UNIVERSITÉ DU QUÉBEC À MONTRÉAL

RECHERCHE D'UNITÉS DE MESURE DANS LES PROTÉINES

RUBAN À MESURER

MÉMOIRE

PRÉSENTÉ

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

NICOLAS MASSOULIER

DECEMBRE 2013

REMERCIEMENTS

Je souhaite dans un premier temps, remercier sincèrement ma directrice Anne Bergeron. Elle a été disponible tout au long de ma maîtrise, son aide et ses conseils ont été précieux notamment durant le travail de rédaction. Elle m'a beaucoup appris, ce qui m'a donné goût à la recherche et à l'envie de faire de nouvelles découvertes scientifiques.

Je voudrais également remercier Bruno Daigle, un ami étudiant en bioinformatique qui m'a permis de passer des moments agréables durant ces années d'études. Il m'a souvent offert son aide et cela a été grandement apprécié.

Enfin, je souhaite dire un grand merci à mes parents Jean-Marc et Caroline, sans qui rien de tout cela n'aurait été possible. Avec ma sœur, Élodie, leur présence et leur soutien sont inestimables et plus forts que la distance qui nous sépare.

L'appui d'Allison a également été très précieux et a contribué à ma réussite.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

RÉSUMÉ

Lors de l'assemblage de la queue d'un bactériophage, une protéine particulière possède la caractéristique de s'étirer, ce qui lui donne un rôle de mesure. Elle est la protéine *ruban à mesurer*.

Les protéines sont codées par des séquences d'ADN et la longueur de la queue du bactério- phage est proportionnelle à la taille de la séquence du *ruban à mesurer*. Comme un vrai ruban à mesurer, les séquences codant ces protéines possèdent des unités répétées.

Biologiquement, des séquences répétées sont appelées des *duplications en tandem*. Ce mémoire a pour objectif d'identifier des *duplications en tandem* au sein des protéines *ruban à mesurer*, qui agissent comme unités de mesure.

Les outils bioinformatiques actuels traitant ces problèmes sont inutiles dans les cas des séquences *ruban à mesurer*. Nous avons donc dû étudier ces séquences à l'aide de logiciels de traitement de chaînes de caractères et de programmes Python. Nos recherches nous ont permis de découvrir 9 motifs d'unités de mesure différents. Ces unités de mesure ont été découvertes au sein de 88 familles de bactériophages, soit un total de 256 séquences. Ces séquences proviennent de notre jeu de données initial, mais également des bases de données du NCBI dans lesquelles nous avons effectué un enrichissement de notre ensemble de données.

MOTS CLÉS : Bioinformatique, bactériophage, duplication en tandem, ruban à mesurer, unité de mesure.

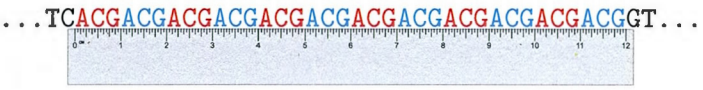
INTRODUCTION

La bioinformatique, ou biologie computationnelle, vient se placer entre l'informatique et la biologie moléculaire. Le séquençage consiste à traduire des molécules biologiques en une succession de quatre caractères qui constituent l'ADN. Suite à l'essor des techniques de séquençage, il a fallu développer des outils informatiques afin d'analyser des génomes, gènes et protéines. Des bases de données ainsi que des algorithmes ont alors été conçus dans le but de déchiffrer les millions de bases composant les séquences d'ADN.

Aux États-Unis, la bibliothèque nationale de médecine abrite le NCBI : centre national pour l'information biotechnologique où sont stockées les séquences biologiques. En août 2013, 167 millions de séquences y sont recensées et représentent approximativement un pétaoctet (soit 1000 téraoctets ou un million de gigaoctets). Comparativement, en février 2010, seuls 10 téraoctets étaient utilisés pour stocker 57 millions de séquences (U.S. National Library of Medicine, 2012).

Parmi ces séquences nous retrouvons un grand nombre de génomes complets, soit toutes les bases qui constituent un organisme dont les bactéries et les virus. Nous nous intéressons au lien entre ces deux derniers puisque certains virus s'attaquent aux bactéries : les *bactériophages*. Ceux-ci sont un assemblage de protéines qui se déplace de manière indépendante afin d'aller infecter une bactérie.

Dans une suite de caractères représentant de l'information génétique plusieurs mutations peuvent survenir (la modification d'un caractère en un autre, par exemple). La *duplication en tandem* est une mutation bien particulière : elle est une succession de copies d'une portion de la séquence (ACG dupliqué en ACGACG par exemple).



À la même manière que 10 millimètres forment un centimètre, une unité de mesure peut apparaître dans une séquence si l'on identifie plusieurs copies successives de même longueur. Cela sera le cas chez les bactériophages dans lesquels une protéine atypique a la particularité de s'étirer : la protéine *ruban à mesurer*.

Le *ruban à mesurer* contrôle la taille de la queue du bactériophage. Cette taille sera déterminée par le nombre de duplications en tandem présentes dans la séquence du *ruban à mesurer*. Le but de ce mémoire est d'identifier le plus grand nombre d'unités de mesure au sein de ces séquences.

Dans le premier chapitre, nous présentons les préliminaires biologiques nécessaires à la bonne compréhension de ce mémoire. Le deuxième chapitre explique l'opération de duplication en tandem et présente les outils bioinformatiques utilisés pour les traiter. Ici sera précisé le lien entre les duplications en tandem et les unités de mesure présentes dans les protéines *ruban à mesurer*. Le chapitre 3 présente nos recherches ainsi que nos résultats. On y trouvera le groupement des séquences par familles, les différents motifs d'unités de mesure identifiés ainsi que leur fréquence d'apparition. Les trois exemples

les plus intéressants y sont présentés. Enfin, le quatrième et dernier chapitre explique comment nous avons enrichi notre jeu de données. Les séquences ajoutées proviennent soit de celles mises à l'écart dans notre ensemble de données initial, soit des bases de données du NCBI. Dans ce dernier cas, nous avons recherché des séquences qui ne sont pas encore annotées *ruban à mesurer*, mais qui contiennent une unité de mesure identifiée. Enfin, nous terminons par un lien avec la biologie en étudiant la relation entre la longueur des *rubans à mesurer* et l'épaisseur de la paroi cellulaire de la bactérie qui est attaquée.

CHAPITRE I

LES BACTÉRIOPHAGES

La plupart des êtres vivants sont composés de cellules. Elles sont classées en deux grands groupes dépendant de leur structure : les eucaryotes et les procaryotes. Les *eucaryotes* sont des cellules de grande taille comportant un noyau et constituent, entre autres, les organismes des végétaux et des animaux. Les *procaryotes* sont des êtres unicellulaires, de plus petite taille et ne possèdent pas de vrai noyau. Les deux grands domaines que l'on retrouve chez les procaryotes sont les *bactéries* et les *archéobactéries.*

Les *virus* sont souvent considérés comme des êtres vivants, mais cette vision est parfois démentie (Villarreal, 2004). Ils sont des organismes composés de protéines et possèdent leur code génétique enfermé en leur centre. À la différence des êtres cellulaires eucaryotes et procaryotes, ils ne sont pas une cellule et ne possèdent pas de noyau. Cependant, leur but est d'infecter une cellule afin de se répliquer, ce qu'ils ne peuvent faire de façon indépendante.

Le but de ce chapitre est d'introduire les notions biologiques nécessaires à la compréhension de ce mémoire. Les bactériophages et le support de l'information génétique y seront explicités.

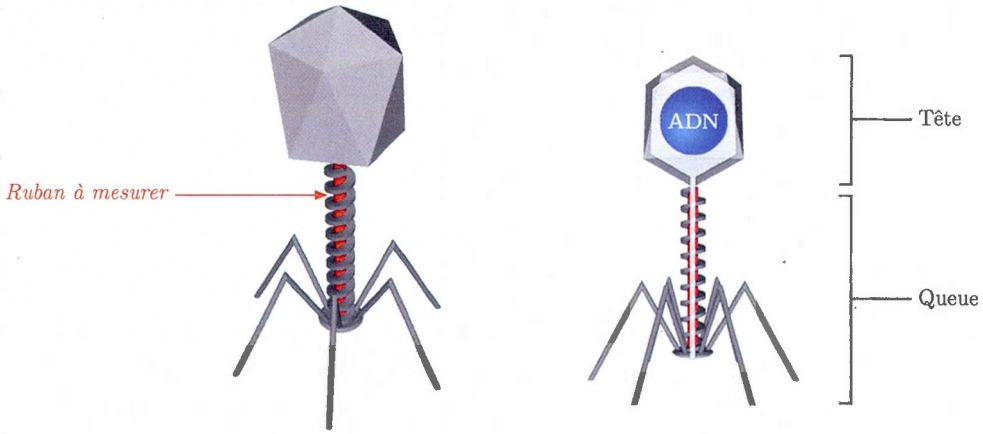


Figure 1.1: Représentation d'un virion de bactériophage et coupe transversale

1.1 Les bactériophages

Les *bactériophages* sont des virus qui infectent les bactéries et archéobactéries. Ils ont été observés pour la première fois en 1915 par le bactériologiste Anglais FW Twort (Twort, 1915). Ses travaux ont ensuite été développés en 1917 par le microbiologiste Franco-Canadien Felix d'Hérelle (d'Hérelle, 1917).

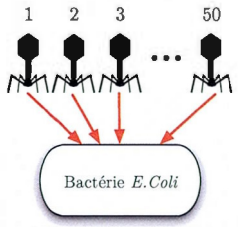
Les bactériophages se sont développés à de nombreux endroits sur la planète et ils sont désormais présents partout dans la nature, et plus particulièrement dans les océans et dans la terre. Ils forment le groupe d'entités biologiques le plus abondant sur la planète. La figure 1.1 donne une représentation des phages les plus communs. Ils sont constitués d'une capside (la tête) qui contient leur génome et d'une queue de longueur variable

selon les espèces. Cette dernière sert à injecter le génome du bactériophage à travers la membrane qui enveloppe la bactérie qui sera infectée.

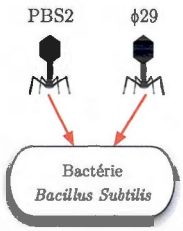
1.1.1 Différents types d'infection

À l'aide des techniques de cultures biologiques et plus tard confirmé par la génomique, il a été démontré qu'il existe trois types d'infections différentes chez les bactériophages :

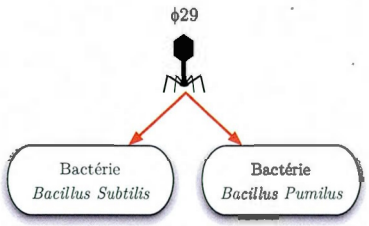
1. **Un grand nombre** de phages peuvent cibler une unique espèce de bactéries. Par exemple, plus de 50 phages différents peuvent infecter la bactérie la plus connue et la plus étudiée : *E. coli* (Buchen-Osmond, 2003).



2. **Quelques** phages peuvent attaquer une même bactérie. Par exemple, seuls les deux phages $PBS2$ et $\phi 29$ sont connus pour attaquer la bactérie *Bacillus Subtilis*.



3. Enfin, **plusieurs** bactéries différentes peuvent être attaquées par le même bactériophage. Par exemple, les bactéries *Bacillus Subtilis* et *Bacillus Pumilus* peuvent être attaquées par le même phage $\phi 29$.



1.1.2 Les phages : nombre d'espèces et d'individus sur la planète

Les différents types d'infection pris en compte, les biologistes se sont entendus pour estimer que le nombre moyen d'espèces de phages pouvant infecter une seule espèce de bactérie est de 10 (Rohwer, 2003).

Il existe aux alentours de dix millions d'espèces bactériennes vivant dans la nature (Curtis, Sloan et Scannell, 2002; Novotny et al., 2002). Ainsi, estimant que chaque espèce de bactérie peut être la cible d'au moins dix espèces de bactériophages, il y aurait alors près de 100 millions d'espèces de phages sur la planète. En terme d'individus, on estime à 10^{30} le nombre de bactéries sur Terre (Whitman, Coleman et Wiebe, 1998), ce qui nous permet d'évaluer le nombre de phages à 10^{31}. Cela fait des phages la forme de vie la plus abondante sur terre.

1.2 L'information génétique

1.2.1 Acide désoxyribonucléique : ADN

L'*acide désoxyribonucléique* (ADN) est le support de l'information génétique. Il est constitué d'une chaîne de *bases*, également appelées *nucléotides* tel qu'illustré en Figure 1.2.

Figure 1.2: Représentation schématisée d'une chaîne d'ADN

Il existe quatre *nucléotides* différents : l'adénosine (A), la thymine (T), la guanine (G) et la cytosine (C). Le sens de lecture de l'ADN est important. Lire une chaîne d'ADN « dans le bon sens » (de gauche à droite) correspond à la lire dans le sens appelé *5' vers 3'* comme présenté en Figure 1.2. L'ADN peut avoir un seul ou deux brins en fonction de l'organisme. Dans un organisme double brin, les bases se lient afin de créer des *paires de bases* et former la bien connue double hélice d'ADN. De plus, les bases ne se lient pas aléatoirement mais selon des propriétés chimiques : l'adénosine se lie à la thymine (A-T) et la guanine se lie à la cytosine (G-C). Il est important de respecter les sens de lecture *5'-3'* sur les deux brins en sachant que les deux brins sont dans des orientations opposées. Un exemple expliquant les sens de lecture ainsi que les liaisons du double brin d'ADN est présenté en Figure 1.3.

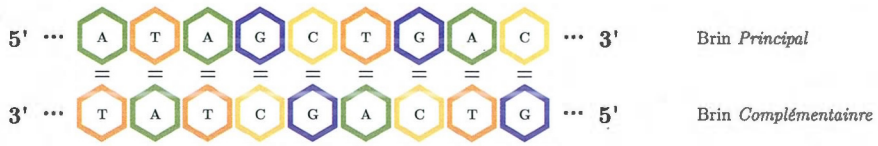


Figure 1.3: Double brin d'ADN

Le brin *principal* se lit ATAGCTGAC, et le brin *complémentaire* : GTCAGCTAT

L'ensemble des molécules d'ADN qui contiennent la totalité de l'information génétique d'une espèce est appelé le *génome*.

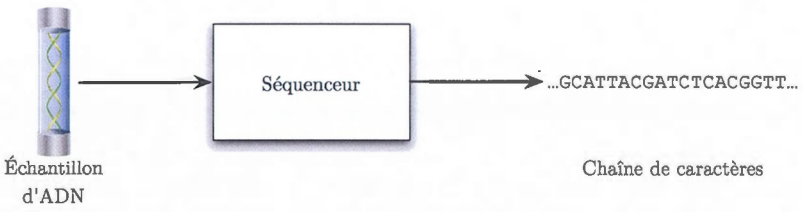
Dans les bases de données biologiques, un seul brin est conservé et les séquences que l'on y trouve correspondent à un seul des deux brins. Cela a simplement un côté pratique car il est facile de retrouver la séquence complémentaire. Il n'y a pas de distinction biologique entre un brin ou l'autre. Une séquence est toujours lue de *5'* à *3'*. Un exemple des différentes possibilités de lecture est présenté ci-dessous :

```
5'-ATGT-3'
3'-TACA-5'
```

La séquence *principale* est lue **ATGT** tandis que la séquence *complémentaire* déduite est **ACAT**. Une seule des deux séquences est conservée dans les bases de données.

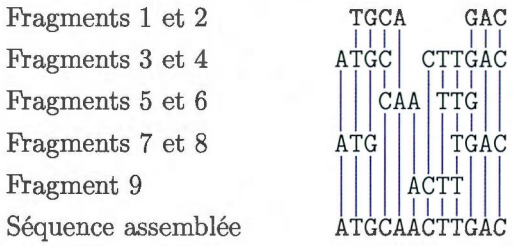
1.2.2 Séquençage

La lecture des molécules d'ADN est appelée le *séquençage*. Les instruments permettant cette lecture sont appelés des *séquenceurs*. En entrée, les séquenceurs ont besoin d'une molécule biologique contenant de l'ADN. En sortie, une simple chaîne de caractères sera retournée. Le premier organisme a avoir été entièrement séquencé est le bactériophage $\phi X174$ par Sanger et al. en 1977 (Sanger et al., 1977).



Une méthode couramment utilisée aujourd'hui est le séquençage global (ou *shotgun sequencing*). Cette technique repose sur un criblage de l'ADN qui retourne de très

courtes séquences, appelées *fragments* (ou *reads*). Une fois ces nombreuses séquences retournées, un problème algorithmique se pose : *l'assemblage.* Il consiste en une superposition minutieuse de ces courts fragments afin de reconstruire la séquence d'ADN initiale. Ci-dessous, un exemple d'assemblage de 9 fragments afin de produire la séquence `ATGCAACTTGAC`.



Selon les différentes techniques de séquençage, la longueur des fragments à assembler varie de quelques dizaines de nucléotides à plusieurs centaines.

1.2.3 Taille des génomes

De nombreux génomes ont été séquencés y compris ceux qui nous intéressent particulièrement : les génomes des phages qui sont relativement petits, mais remplis d'informations. Une étude a démontré qu'un génome de bactériophage a une taille moyenne de 50 000 nucléotides (Breitbart et al., 2002).

En comparaison, au 7 mai 2013, le génome de l'humain *(homo sapiens)* en version 37.10 a une taille de 3 224 463 819 nucléotides.

1.3 Bases de données biologiques

À la fin des années 80, les États-Unis ont mis en place le *NCBI : National Center for Biotechnology Information* (U.S. National Library of Medicine, 2012), qui est une base de données recensant toutes les données biologiques séquencées. Ainsi, depuis plus de vingt ans, toutes les informations extraites par les biologistes ont été stockées au NCBI, qui est gratuit et accessible à tous. Ainsi, au 30 Avril 2013, 730 phages ont été séquencés, dont 692 infectent les bactéries et 38 les archéobactéries. Si l'on fait l'analogie avec les 100 millions d'espèces de phages sur Terre, seules 0,000007% des espèces sont séquencées.

1.4 Transcription et traduction

Comme vu précédemment, l'ADN est le support de l'information génétique. Cependant, celui-ci doit être utilisé et transformé afin de créer les composantes des êtres vivants. Pour y arriver, des étapes biologiques appelées *transcription* et *traduction* sont nécessaires.

1.4.1 Transcription

La première étape consiste en la création d'une première copie du brin d'ADN appelée le transcrit. Celui-ci est la transformation du brin d'ADN en *acide ribonucléique messager* (ARNm). Au cours du processus, les bases de thymine (T) sont converties en bases d'uracile (U), spécifiques à l'ARNm.

Cette opération est appelée la *transcription*. Un exemple est présenté ci-dessous.

$$\texttt{ATGTTATCCCAC} \xrightarrow{transcription} \texttt{AUGUUAUCCCAC}$$

1.4.2 Traduction

Le brin d'ARN messager subit ensuite la prochaine étape de synthèse. Des molécules viennent se lier sur l'ARNm afin de traiter son information pour la traduire en *acides aminés* : ceux-ci, par leur assemblage, constituent les protéines. Les génomes des organismes vivants peuvent coder 22 acides aminés différents.

Les molécules qui se lient sur l'ARNm ont la spécificité de lire les nucléotides trois par trois. Une lecture de trois nucléotides produit un acide aminé. Cette étape est appelée la *traduction*. Chaque chaine de trois nucléotides est traduite par un seul et unique acide aminé. Un code de traduction a donc été déterminé biologiquement : il est le *code génétique standard* (Elzanowski et Ostell, 2012), donné en Tableau 1.1. Les abréviations des acides aminés sont données en Tableau 1.2.

Un exemple de traduction est donné ci-dessous :

ATG	TTA	TCC	CAC	AAT	AGA	AAG	GCT	TAA
M	L	S	H	N	R	K	A	STOP

	T		C		A		G	
T	TTT	F	TCT	S	TAT	Y	TGT	C
	TTC	F	TCC	S	TAC	Y	TGC	C
	TTA	L	TCA	S	TAA	stop	TGA	stop
	TTG	L	TCG	S	TAG	stop	TGG	W
C	CTT	L	CCT	P	CAT	H	CGT	R
	CTC	L	CCC	P	CAC	H	CGC	R
	CTA	L	CCA	P	CAA	Q	CGA	R
	CTG	L	CCG	P	CAG	Q	CGG	R
A	ATT	I	ACT	T	AAT	N	AGT	S
	ATC	I	ACC	T	AAC	N	AGC	S
	ATA	I	ACA	T	AAA	K	AGA	R
	ATG	M	ACG	T	AAG	K	AGG	R
G	GTT	V	GCT	A	GAT	D	GGT	G
	GTC	V	GCC	A	GAC	D	GGC	G
	GTA	V	GCA	A	GAA	E	GGA	G
	GTG	V	GCG	A	GAG	E	GGG	G

Tableau 1.1: Code génétique standard

Code	Abréviation	Acide aminé
A	Ala	Alanine
C	Cys	Cystéine
D	Asp	Acide aspartique
E	Glu	Acide glutamique
F	Phe	Phénylalanine
G	Gly	Glycine
H	His	Histidine
I	Ile	Isoleucine
K	Lys	Lysine
L	Leu	Leucine
M	Met	Méthionine
N	Asn	Asparagine
O	Pyl	Pyrrolysine
P	Pro	Proline
Q	Gln	Glutamine
R	Arg	Arginine
S	Ser	Sérine
T	Thr	Thréonine
U	Sec	Sélénocystéine
V	Val	Valine
W	Trp	Tryptophane
Y	Tyr	Tyrosine

Tableau 1.2: Acides aminés et leurs abréviations

1.4.3 Cadres de lecture

Lors de la traduction, les nucléotides de l'ARNm sont lus trois par trois. Une chaine de trois bases est plus communément appelée un *codon*. Le code génétique est basé sur une série de codons et présente donc certaines contraintes. Celles-ci sont liées au respect d'un cadre de lecture car décaler la lecture de l'ARNm d'un nucléotide aura comme répercussion la lecture d'un codon totalement différent. Cette contrainte est illustrée en Figure 1.4

Figure 1.4: Exemple de trois différents cadres de lecture lors de la traduction de l'ARNm en acides aminés. La même suite de nucléotides peut donc donner 3 suites différentes d'acides aminés

1.5 Phases de la vie d'un phage

Un phage possède différentes phases. La phase vue jusqu'à présent concerne la forme inactive du virus appelée le *virion*. Il est un moyen de transport autonome pour l'ADN viral afin de se déplacer pour infecter une bactérie.

Une fois en contact avec une bactérie, le mécanisme d'infection se met en place. Il est composé de quatre étapes :

1. **Infection** Le virion du bactériophage vient se fixer sur une bactérie. Il injectera ensuite son ADN viral à l'intérieur de la bactérie en perforant sa membrane.
2. **Réplication** Au sein de la bactérie, l'ADN viral utilisera la machinerie cellulaire afin de se répliquer. Un seul brin d'ADN injecté par le phage va rapidement se répliquer et synthétiser un grand nombre de protéines composant les bactériophages.
3. **Assemblage** Les nombreuses copies des protéines synthétisées vont s'assembler au sein de la bactérie afin de créer de nouveaux virions.
4. **Évasion** Une horloge biologique enverra un signal afin de détruire la bactérie (Wang, Smith et Young, 2000). Sa paroi cellulaire se déchirera et elle mourra. Les nouveaux phages relâchés pourront désormais infecter d'autres bactéries (retour à l'étape 1).

Ce processus est appelé l'infection *lytique*. Cependant, au cours de ce cycle, il se peut que l'ADN viral du phage vienne s'insérer dans l'ADN de la bactérie. Le processus d'infection rentre alors dans le cycle *lysogénique*.

L'étape d'**infection** est similaire. Cependant, l'ADN viral n'utilisera pas la machinerie cellulaire afin de se répliquer, mais il s'insérera dans l'ADN de la bactérie ; le virus devient alors un *prophage*.

Un *prophage* est une portion d'ADN viral, au sein d'un génome de bactérie ayant été infectée par un bactériophage. Il peut y avoir plusieurs *prophages* au sein d'une même bactérie et un exemple est proposé en Figure 1.5

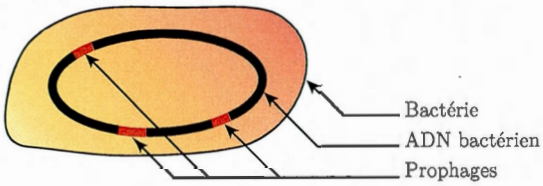


Figure 1.5: Représentation schématisée d'une bactérie et mise en évidence des prophages

La bactérie étant une cellule, elle va subir la division cellulaire et ainsi créer des copies du prophage. Les prophages à l'état passif peuvent retourner dans le cycle lytique suite à un événement déclencheur. L'ADN viral va alors se détacher de l'ADN bactérien et ainsi retourner à l'étape 2 : **réplication** (Alberts, 1995).

La Figure 1.6 résume ces deux cycles.

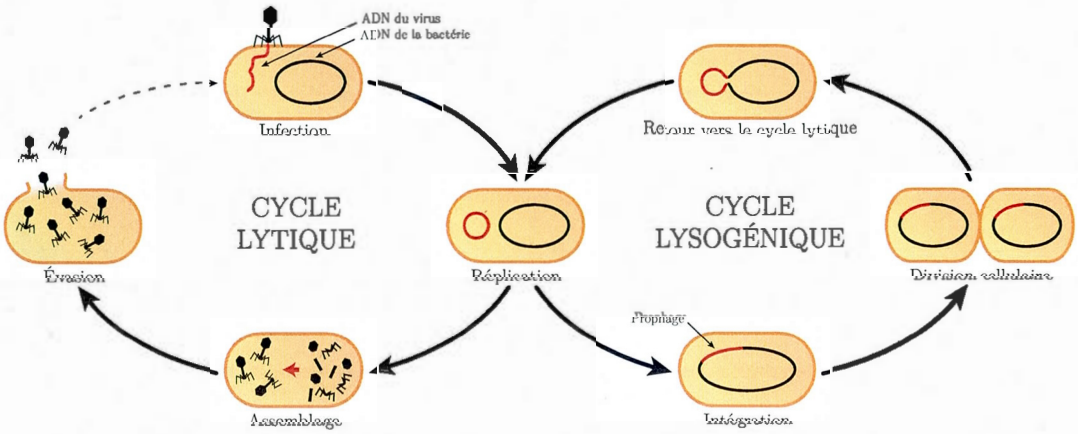


Figure 1.6: Infection d'une bactérie : cycle lytique et cycle lysogénique

Cette différence (phage-prophage) est très importante au niveau bioinformatique. Afin de trouver des génomes de phages nous avons donc le choix de chercher dans les génomes de virus ou bien dans les génomes de bactéries. Premièrement, cela présente une différence dans le choix de la base de données à utiliser. Deuxièmement, une étude

au sein d'un bactériophage est moins complexe car nous avons vu qu'ils ont une taille moyenne de 50 000 nucléotidès, contrairement aux bactéries qui ont des tailles variables de quelques millions de nucléotides (par exemple, la bactérie *Escherichia Coli* a une taille de 5 millions de nucléotides (Blattner et al., 1997)). Ainsi, étudier la portion d'ADN appartenant au virus (le prophage) au sein d'une bactérie est bien plus complexe que directement dans le génome du phage.

PHAST *(PHAge Search Tool)* (Zhou et al., 2011) est un outil de recherche de prophages dans les génomes de bactéries. Sa base de données est basée sur des séquences de phages et de prophages issues du NCBI et de ProphageDB (Srividhya et al., 2006), une base de données de prophages. Ainsi, avec un génome de bactérie en entrée, PHAST cherche si des portions de ce génome sont similaires à des prophages existants. En sortie, on peut observer les prophages identifiés ainsi que leur pourcentage de conservation par rapport au prophage dans la base de données. Ils sont alors annotés « intact », « douteux » ou « incomplet ».

1.6 Rubans à mesurer

Une des protéines liées à la phase d'assemblage des virions est codée par une séquence dont la longueur est proportionnelle à la taille de la queue du virion : elle agit donc comme un ruban à mesurer (Katsura et Hendrix, 1984).

À la manière des vrais rubans à mesurer, les séquences qui codent pour ces protéines contiennent des unités répétées. Le *ruban à mesurer* est représenté en rouge dans la Figure 1.1

L'objectif de ce mémoire est de découvrir et d'analyser ces unités répétées, de façon à pouvoir retracer une partie de l'histoire de l'évolution des nombreuses espèces de phages qui peuplent la planète.

Le prochain chapitre portera sur les outils mathématiques et informatiques pour l'étude des duplications en tandem dans les séquences.

CHAPITRE II

DUPLICATIONS EN TANDEM

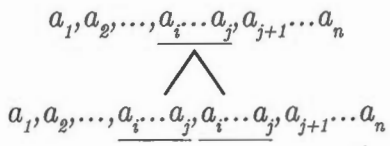
Au sein des séquences d'ADN ou d'acides aminés, des unités de mesure peuvent être identifiées sous forme de séquences répétées. Au niveau moléculaire cela se traduit par des répétitions de morceaux de séquence de longueurs fixe (ou presque fixe). Informatiquement, il s'agit de trouver des répétitions de chaînes de caractères successives de longueur constante. Ce chapitre présente l'opération de duplication et détaille les difficultés qui y sont liées. À savoir, le problème des frontières, les mutations au sein des duplications et la variabilité de la période.

2.1 Opérations de duplication

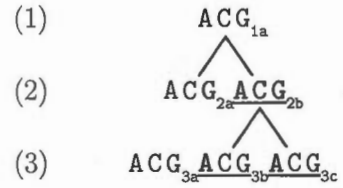
Formellement, étant donné une séquence de nucléotides $a_1, a_2, \ldots, \underline{a_i \ldots a_j}, a_{j+1} \ldots a_n$, une *duplication en tandem* des nucléotides $a_i \ldots a_j$ est la production de la séquence : $a_1, a_2, \ldots, \underline{a_i \ldots a_j}, \underline{a_i \ldots a_j}, a_{j+1} \ldots a_n$.

La longueur de la duplication est appelée *la période*. Elle est de taille $j - i + 1$.

On représente cette opération par un arbre :



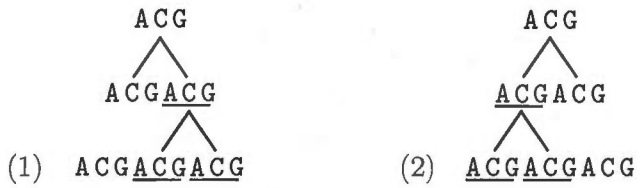
Plusieurs duplications peuvent se produire dans la même séquence. Par exemple :



Cet arbre nous renseigne sur les duplications successives de la séquence de base **ACG** afin qu'elle devienne **ACG ACG ACG**. Ici, c'est la séquence $\mathbf{ACG}_{1a}$ qui s'est dupliquée en premier. Ensuite, $\mathbf{ACG}_{2b}$ s'est dupliquée à son tour afin de créer la séquence au niveau (3). La séquence $\mathbf{ACG}_{1a}$ est appelée *séquence ancestrale.*

Cet exemple présente la manière dont les séquences se dupliquent, nous les avons illustrées de « haut en bas ». Un problème central en bioinformatique est la reconstruction des duplications successives qui ont mené à une séquence particulière. En effet, nous ne disposons pas, en général, des séquences ancestrales qui se sont souvent perdues au cours de l'évolution.

Ainsi, si l'on part de la séquence **ACG ACG ACG**, on observe que l'arbre vu précédemment était juste, mais qu'un autre arbre peut aussi expliquer la même séquence finale. Les deux résultats possibles (qui mènent aux mêmes séquences finales) sont présentés ci-dessous.



En possédant uniquement **ACG ACG ACG**, il est impossible de savoir si la reconstruction est (1) ou (2). Avec cette simple succession de duplications, on ne peut pas savoir laquelle est survenue en premier et nous sommes dans l'incapacité de choisir un des arbres. Le problème devient encore plus complexe lorsqu'on considère la variabilité des frontières.

2.2 Frontières fixes et frontières variables

Les *frontières* sont les endroits qui délimitent le début et la fin d'un événement de duplication. Elles peuvent être fixes ou variables.

Par exemple, considérons un événement de duplication survenu sur la séquence **ACGT**. Les frontières fixes de cette duplication se situeront toujours entre **A** et **T**.

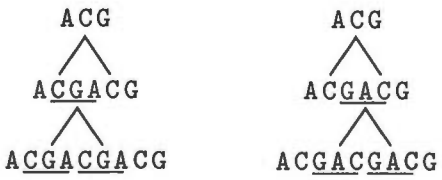
$$\begin{aligned} \underline{\texttt{ACGT}} &\rightarrow \texttt{ACGT}\,\underline{\texttt{ACGT}} && (1)\\ &\rightarrow \underline{\texttt{ACGTACGT}}\,\texttt{ACGT} && (2)\\ &\rightarrow \texttt{ACGTACGT ACGTACGT ACGT} && (3) \end{aligned}$$

Considérons ce même exemple, mais avec des frontières variables. C'est-à-dire que les événements de duplications ne surviendront pas uniquement entre **A** et **T**, mais, à n'importe quel endroit dans la séquence. Par exemple, ici, entre **G** et **C** puis, entre **T** et **G** :

ACGT → AC GTAC GT (1)
→ ACG TACGTACG T (2)
→ ACG TACGTACG TACGTACG T (3)

On observe alors qu'il est possible d'obtenir le même résultat (les deux séquences de niveau (3) sont identiques) avec des frontières différentes. Cependant, les deux séquences n'ont pas du tout la même histoire.

Appliquons le modèle de frontières variables à la séquence précédente : ACG ACG ACG. Aux deux arbres précédents, nous pouvons désormais en ajouter deux de plus. La séquence de niveau (2) pourrait alors se dupliquer au niveau de CGA (ACGACG) ou encore au niveau de GAC (ACGACG). Cela complique grandement le problème de la reconstruction de l'histoire des duplications. Ces arbres supplémentaires sont présentés ci-dessous.



Si l'on considère uniquement le modèle à frontières fixes, l'histoire des événements de duplication peut être reconstruite à l'aide d'*arbres de duplication*. Il s'agit d'arbres enracinés qui ont été déracinés afin de mieux observer l'histoire des duplications. Les *feuilles* ordonnées représentent les duplications (Elemento, Gascuel et Lefranc, 2002).

En bioinformatique, il existe de nombreux outils phylogénétiques qui traitent ce type de duplications. Cependant, le modèle à frontières variables est bien plus complexe, les

arbres de duplication ne suffisent pas et les outils développés pour les frontières fixes ne sont pas adaptés à ce type de problème (Belcaid, Bergeron et Poisson, 2011).

Les duplications que nous étudions au sein des protéines *ruban à mesurer* sont à frontières variables.

2.3 Duplications et mutations

Nous avons vu que les différents scénarios possibles pour la reconstruction des duplications en tandem sont nombreux. Jusque là, il nous a été impossible de déduire avec certitude l'historique des duplications, notamment la *plus récente duplication.* Cependant, les séquences biologiques sont différentes des duplications *parfaites* que nous avons vues jusqu'ici. Nous trouvons, au sein de ces séquences, des *mutations* qui permettent, parfois, de retracer l'évolution des différentes séquences dupliquées.

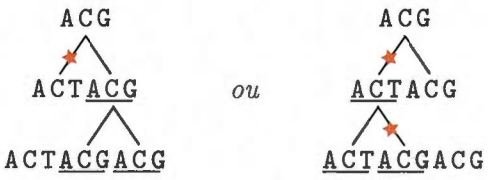
Une *mutation* est la modification d'un nucléotide. Ces mutations peuvent être de différentes natures, mais nous allons, pour le moment, parler uniquement de la *substitution* qui est la mutation d'une base en une autre. Dans d'autres termes, une lettre sera remplacée par une autre dans la séquence.

Un exemple de substitution est présenté ci-dessous où `G` a muté en `A` :

`ATGATG`
`ATAATG`

Considérons maintenant la séquence mutante `ACTACGACG`, qui est une variation de notre exemple précédent. Cette mutation ajoute une information très importante pour la reconstruction. L'étoile représente une mutation entre la séquence parente et l'une de ses

deux filles. Voici deux reconstructions – à frontières fixes – pour la séquence **ACTACGACG** :



Ces deux arbres mènent au même résultat, mais avec un nombre de mutations différent. Afin de faire un choix, nous allons utiliser la *parsimonie.*

Le principe de *parsimonie* repose sur les arbres de Wagner (Wagner, 1961) et a ensuite été explicité par Fitch (Fitch, 1971). Selon lui, un arbre présentant le *maximum de parsimonie* est celui qui a connu le moins de changement durant l'évolution. Autrement dit, un *meilleur* arbre sera celui qui présente le moins de mutations. Dans notre exemple ci-dessus, le *meilleur* arbre est celui avec une seule mutation.

Afin de trouver les reconstructions qui possèdent le moins de duplications, nous avons besoin d'étudier leurs *distances.*

2.3.1 Distance entre duplications

La *distance* est la mesure de l'éloignement entre deux séquences ; elle est toujours positive ou nulle.

Une distance populaire en informatique est la *distance de Hamming* (Hamming, 1950). Hamming a défini cette distance comme étant le nombre d'éléments qui diffèrent entre deux chaînes de caractères de même longueur $\mathbf{x}$ et $\mathbf{y}$ et est notée $h(\mathbf{x},\mathbf{y})$.

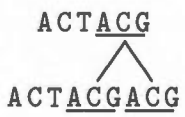
Dans notre exemple précédent, nous avions la séquence **ACTACGACG**. Avec une *période* = 3 ainsi que des frontières variables, nous avons quatre duplications candidates à étudier, exposées ci-dessous :

(1) ACT ACG ACG
(2) A CTA CGA CG
(3) AC TAC GAC G
(4) ACT ACG ACG

Les distances de Hamming entre ces paires de séquences sont les suivantes :

(1) $h(\mathtt{ACT}, \mathtt{ACG}) = 1$
(2) $h(\mathtt{CTA}, \mathtt{CGA}) = 1$
(3) $h(\mathtt{TAC}, \mathtt{GAC}) = 1$
(4) $h(\mathtt{ACG}, \mathtt{ACG}) = 0$

La duplication la plus récente est souvent celle qui a la plus faible distance. Le scénario (4) est celui qui présente la distance la plus faible entre deux duplications. La distance entre **ACG** et **ACG** étant nulle, elle est la plus récente duplication nous permettant de reconstruire un niveau de l'arbre :



Ce simple exemple présente une période de longueur 3. Expérimentalement, nous allons rencontrer des répétitions avec une période bien plus grande.

2.3.2 Variation de la période

Le calcul des distances entre les duplications devient de plus en plus difficile à raison que la période augmente. De plus, la variation des frontières apporte une difficulté supplémentaire.

Prenons pour exemple, la séquence **ACGT ACTG ACGT ACTA**. Visuellement, sur cette courte séquence, nous observons rapidement une période de 4 et, après calcul des distances, nous trouvons que chaque duplication se trouve à une distance de 2.

(1) ACGT ACTG ACGT ACTA
(2) ACGT ACTG ACGT ACTA
(3) ACGT ACTG ACGT ACTA

(1) $h(\texttt{ACGT},\texttt{ACTG}) = 2$
(2) $h(\texttt{ACTG},\texttt{ACGT}) = 2$
(3) $h(\texttt{ACGT},\texttt{ACTA}) = 2$

Cependant, si l'on étudie les différentes périodes possibles dans cette séquence, la période 8 attire particulièrement l'attention. En effet, elle est dans cet exemple la duplication présentant la plus faible distance (distance = 1) et donc, la plus récente duplication.

ACGTACTG ACGTACTA

$h(\texttt{ACGTACTG},\texttt{ACGTACTA}) = 1$

Cet exemple permet de mesurer la complexité du problème des duplications en tandem.

Les outils informatiques sont donc indispensables afin de calculer la totalité des distances pour toutes les combinaisons de *frontières/périodes*.

La seule approche connue (Rivals, 2004) traitant le problème des duplications en tandem à frontières variables est l'heuristique de Gary Benson : *Tandem repeats finder* (Benson, 1999).

2.4 Outils bioinformatiques

2.4.1 *Tandem Repeats Finder*

Tandem Repeats Finder, développé par Gary Benson est une heuristique de recherche de duplications en tandem à frontières variables dans une séquence de nucléotides.

Le programme est basé sur un module de *détection* qui recherche des duplications potentiellement candidates. Ensuite, le module d'*analyse* tente d'aligner chaque candidat afin de retourner des informations statistiques sur la qualité de la duplication.

Module de détection. L'algorithme commence par chercher des groupements k, appelés des k–uplets. Un k–uplet correspond alors à une fenêtre sur la séquence et des k–uplets similaires sont deux fenêtres contenant les mêmes séquences. Par exemple si $k = 5$, deux k–uplets similaires seront `ACTGC` et `ACTGC`.

Le nombre de séquences possibles, à l'intérieur des fenêtres, avec l'alphabet de nucléotides (`A-C-G-T`) est 4^k. Ces séquences sont appelées des *sondes p*.

Les positions i de chaque sonde sont enregistrées dans un historique H_p. Ainsi, pour chaque position i ajoutée à H_p, l'algorithme inspecte H_p pour toutes les occurrences

précédentes de cette sonde. Si une position précédente de cette sonde est j alors, la distance $d = i - j$ est une taille possible pour une duplication en tandem. Cette distance d est stockée dans une liste D_d et cela, à chaque fois qu'une sonde similaire est détectée à une distance d. Si les informations stockées dans cette liste valident les critères de l'utilisateur, la répétition candidate est envoyée au module d'*analyse*.

Module d'analyse. Possédant la liste des positions, un candidat pouvant devenir une duplication en tandem se trouve à la position $j+1 \ldots i$. Un consensus de ce candidat est créé afin de prendre en compte certaines mutations biologiques qui ont pu avoir lieu. Ce consensus est aligné sur la séquence de base et si au mois deux copies s'alignent, le candidat est validé comme étant une duplication en tandem.

2.4.2 Recherche de motifs : *MEME*

Un *motif* est un modèle de séquence qui revient à plusieurs reprises au sein d'une séquence de nucléotides ou d'acides aminés. MEME (Bailey et Elkan, 1994) est un programme bioinformatique de recherche de motifs.

Il recherche des motifs récurrents dont la distance peut varier en fonction des occurrences. La taille de ces motifs est paramétrable. En retour, une matrice de poids score-position (*PSSM : Position-specific scoring matrix*) est présentée au format LOGO (Schneider et Stephens, 1990). Ce format permet de visualiser rapidement les domaines conservés dans un motif.

Par exemple, MEME trouve un motif revenant trois fois au sein d'une protéine *ruban à mesurer* (dont le numéro d'accession est `ZP_06557477.1`).

`NDKAFASASA` **`WVRIQR`** `KLYEENGTDS`
`LDKIKNNTKG` **`WNDEQR`** `SSAIALAFGT`
`PLALPKGTKI` **`WSSMAR`** `FNAHKIFAKS`

Ce motif est bien mieux observable au format LOGO, donné en Figure 2.1. La hauteur des caractères reflète le score de chaque position. Il est le *maximum d'entropie*, premièrement défini par Edwin Jaynes (Jaynes, 1957) et représente la probabilité de chaque acide aminé de survenir à chaque occurrence du motif. Ainsi, `W` et `R` sont présents aux mêmes positions dans les trois occurrences du motif. De plus, à la position 5 nous observons une incertitude entre le `Q` et le `A`, avec une plus grande probabilité pour le **`Q`**.

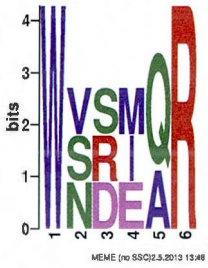


Figure 2.1: Exemple de motif retourné par MEME, au format LOGO

Le prochain chapitre porte sur la recherche des duplications au sein des séquences *ruban à mesurer*. Il présente également les résultats obtenus lors de ces recherches.

CHAPITRE III

DUPLICATIONS EN TANDEM ET UNITÉS DE MESURE DANS LES PROTÉINES *RUBAN À MESURER*

Ce chapitre présente la méthode de construction de notre jeu de données, composé de toutes les séquences *ruban à mesurer* présentes au NCBI. Ce jeu de données a ensuite subi plusieurs traitements afin qu'il soit le plus fiable possible. La suppression des doublons et l'évaluation de l'annotation « *ruban à mesurer* » font partie de ces traitements. Les séquences retenues ont ensuite été groupées par similarité afin de voir apparaître des familles d'unités de mesure en fonction de l'espèce du bactériophage. Ce chapitre expose également les recherches des duplications en tandem afin d'identifier des unités de mesure. Les outils bioinformatiques actuels ne trouvent aucune duplication dans les *rubans à mesurer,* mais des recherches antérieures ont montré un lien entre l'acide aminé tryptophane et l'unité de mesure. Cela nous a permis d'identifier des duplications de taille fixe à l'aide de programmes Python et de logiciels de traitement de chaînes de caractères. Nous avons identifié 18 familles intéressantes qui représentent un total de 129 séquences. Dans ces 18 familles, nous avons découvert 9 motifs d'unité de mesure. Nous discutons en détail des 3 familles les plus intéressantes.

3.1 Format *ProSite*

Au cours de ce chapitre, nous évoquerons les différents motifs d'unités de mesure identifiés dans les séquences *ruban à mesurer*. Afin de traiter convenablement ce type de duplications comportant des motifs alternatifs et des périodes variables, nous avons besoin d'un format pour les schématiser : nous utilisons le format de motifs *ProSite*.

Le format *ProSite* est un langage de description de séquences d'acides aminés. Le symbole « `x` » est utilisé pour une position où n'importe quel acide aminé peut être utilisé. Lorsque plusieurs acides aminés sont acceptables à une même position, on représente ceux-ci entre crochets « `[ ]` ». Par exemple `[FW]`, signifie : phénylalanine (`F`) ou tryptophane (`W`).

Si le motif consiste en une suite d'acides aminés, ils sont notés côte à côte. Sinon, les éléments sont séparés par le symbole « - ». Par exemple, une chaîne méthionine-leucine-proline-valine notée `M-L-P-V` peut être simplifiée par `MLPV`. Sinon, une chaîne composée d'une sérine (`S`), suivie par n'importe quel acide aminé, suivi par une glycine (`G`) est notée : `S-x-G`.

La répétition d'un élément est représentée par l'élément, suivi du nombre de fois qu'il est répété, entre parenthèses ; par exemple, `M-M-M-A-A` se simplifie par `M(3)-A(2)`. S'il y a un nombre d'occurrences variable à partir d'une position donnée, on utilise un intervalle entre parenthèses. Par exemple, « Une cystéine, suivie de 8 à 11 acides aminés » se traduit par : `C-x(8,11)`.

3.2 Récupération des séquences annotées *ruban à mesurer*

La récupération de la totalité des séquences annotées *ruban à mesurer* au NCBI repose sur l'interface de programmation *Utilitaires Entrez* du NCBI. Les séquences ont été récupérées dans la base de données de séquences d'acides aminés appelée *non-redundant*, qui regroupe l'ensemble des séquences répertoriées, peu importe leur origine : génomes complets ou séquençage d'ARN messager, par exemple. Les résultats de la recherche ont été restreints aux annotations incluant les termes « *tape* » et « *measure* ». Nous obtenons alors des séquences d'acides aminés en format FASTA [1] (cette technique est décrite en Appendice A.1).

Lors de la création de notre jeu de données le 5 Avril 2011, nous avions récupéré 6739 séquences d'acides aminés annotées *ruban à mesurer*. Les séquences biologiques associées à ces protéines sont en expansion constante et rapide dans les bases de données du NCBI. À titre de comparaison, le 11 Juin 2013 on en compte désormais 28 054. Lors de nos recherches ainsi que dans ce mémoire, les séquences utilisées proviennent de notre jeu de données initial de 6739 séquences.

Un exemple de séquence retournée est donné en Figure 3.1. On retrouve sur la première ligne, son numéro d'accession suivi de l'annotation *phage tail tape measure protein.* L'espèce de la bactérie attaquée par le phage est *Lactobacillus antri DSM 16041.*

Le jeu de données ainsi créé comporte des séquences à éliminer (séquences trop courtes

1. Voir Appendice C.1

```
>gi|259503473|ref|ZP_05746375.1| phage tail tape measure protein
[Lactobacillus antri DSM 16041]

MAGSSLGHLAATVSLNINPFKASNNVLKAEIKSTANALRAQEIALKGSANSLNNMRAVYATMQSQMNNYN
AQLQRSKALMDDTTRSERSRLNAANQFNKTSAQVEILRTRMQALSREIVAQSSRWGQLAARANHFGNVAT
AVGSRVQGVGRGMSTYLTAPIAAGLAYSASKLVDFQDAMNRTKNVIRTSGESAAETQRSYNQMLRDSRKY
SDMYGVSQIKIANGYQDLVKRGYTSKAAVGVMRNELKASIATGDDFNDVIKVASQTMESFGLSTNKAGQP
LKSARLMQERSKDTLNKLAYAADATSTDFNSLGVGMSYVGATAHQAGFGLGETASAMGILSNNGLEADKA
GTGLRKVIQSLITPTKGGAEALAKINLSTKDFIGKNGKLKSMSTIFATLNKHMKGLTGHEKNDIFHSLFG
TTGQQAGAILTENAKRLRELTNEVNNAGKRDYIGDLAKRNLNTPKAQLAIFKESLTNAGMDAAKNVLPII
TPLVQDLSKLAQGFGQLPEPMKKAITYTTLFVAGAGPLLLLIGKLSSGVGGLALKLGSFAGGMSRARAAI
SMGAGGLTVLKSAFSKNAFETAKFGKAATTASGAMTTLKGTVDTANGTISSSKTIVDSAGRALTTAGESA
AVAGTSFGAAAIPIAAATAVIIAGVAAWELWGKKAAASDERTNRWGSDVGVAADHALSKMQSTSQGIQTA
LSDMDVAAHTSTKKMADDFDREFTQIERSAREHFKNVKKAEKGLSPEVSKAIDAEAEKEKSSFNGYVHDA
DVANQNVQKILSGRNRKVSELDQTQRTMLRNYQAEMLNDEMKTLQLSGSRRAKAMAVLNNDIKHMSQQQR
STRMGDITSETAEMEQQYKRQASLLKKRYKNNELSRAEYRAGLKSNQRALSDYSNKAAAEYIRLARANGE
STSLIKKDLKSMGLSYKAGTAEIRRQTKAAMQDQKSLAVNTTKMSGKVKKAANMWNNLVFDPKTGKVRTN
AQEEVNKAVNSSKGWNQIKLLKKEGKLSTNAQHMVAAALIANGQWDSMSWPDKQAWLHDKFHQTIVKALE
DSGQWNNLTLEQKEAIVNAKGKQEMADLLMESVGWNNLTLQEKQAIVADKATLPLVDSLQKSGQWNGMTL
KQQEAIINAKGKDQLVDALVKGGVWNSLSLKQQLALVRTQGTEQVIQGIDQMGRWNSLSPKQQQAIVSAK
GGPDLQQLITDYGLWRGLPASAAKQIIAQDRASGNLKAANDAMQAWIKANPGAPKNALAIDNASGPLHNA
TGGVNVFAGSSTGAPKVASGVDNASGPFGNATNSIFGWNRTGANTKNIRANDQASKNAQIAISAIDAWNK
AHPIIHKFITEVITKHKNKANGANYFEGGPVMVNDQKGPIFREMVQFPGEIPFIPYGRNVLLNAPRGTRI
VKASDTLKQFPHLPQYANGNSDAVSVLNNIQPVQVSAKGTNDSQSASTSDNKYMGEVLERLGKMTDRLGT
MLGLNAAQLSAIKASAFDKNDLYSKMGMDQVYYDAQRL
```

Figure 3.1: Séquence annotée *tape measure* en format FASTA dont le numéro d'accession est ZP_05746375.1. Cette séquence a comme annotation *phage tail tape measure protein* et elle provient de l'espèce de bactérie *Lactobacillus antri DSM 16041.*

et doublons). Ce traitement a été effectué à l'aide de l'utilitaire CD-Hit (Li et Godzik, 2006) qui est utilisé pour classer des séquences en groupes similaires. Dans notre cas, nous l'utilisons afin de créer des groupements de séquences grandement semblables et ainsi supprimer les doublons. Le détail du traitement est expliqué en Appendice A.2.

Deux séquences peuvent être identiques, mais annotées différemment. Ainsi, après suppression des doublons, notre jeu de données compte désormais 2633 séquences uniques. Nous avons également écarté les séquences dont la taille est inférieure à 200 acides

aminés, celles-ci étant trop courtes pour être étudiées, cela réduisant notre nombre de séquences à 2459.

Les *rubans à mesurer* se retrouvent uniquement dans des phages ou des prophages de bactérie ainsi, nous avons conservé uniquement les séquences dont l'organisme est *Virus* ou *Bactérie* (et *Archéobactérie*). Finalement, nous avons 2434 séquences dans notre jeu de données.

Nous avons épuré notre jeu de données afin que celui-ci soit le plus fiable possible. Cependant, les annotations dans les bases de données du NCBI sont parfois douteuses et nous ne sommes pas certains que l'annotation *ruban à mesurer* corresponde assurément à ce type de protéine. Nous avons donc besoin d'évaluer cette annotation et ce problème est traité dans la section suivante.

3.3 Évaluation de l'annotation

Évaluer l'annotation *ruban à mesurer* nous permet de juger si oui ou non, les séquences correspondent à ce type de protéine. Cette étape nous permet d'augmenter le degré de confiance dans les annotations et permet également d'écarter des séquences dont l'annotation est douteuse. Sans cette étape, il se pourrait que nous cherchions des unités de mesure dans des séquences qui n'en possèdent pas, car elles ne sont pas réellement des *rubans à mesurer*. Elles auraient uniquement une fausse annotation et une provenance tout autre.

Pour valider les séquences, nous disposons de deux caractéristiques propres aux phages et à leur *ruban à mesurer* :

1. **La séquence provient d'un génome de bactériophage.** Un génome de bactériophage a une taille moyenne avoisinant les 50 000 nucléotides [2] (Breitbart et al., 2002), ce qui est considéré comme petit. Dans notre jeu de données, les séquences *ruban à mesurer* ont une taille moyenne de 948 acides aminés, soit 2844 nucléotides : une très grande protéine au sein d'un génome de bactériophage. Grâce à différentes études (Pell et al., 2009; Katsura et Hendrix, 1984; Siponen et al., 2009), nous savons que le *ruban à mesurer* est la plus grosse protéine au sein des génomes de bactériophages.

   Par exemple, l'équipe de Marina Siponen (Siponen et al., 2009) a mis en évidence l'existence de la protéine *ruban à mesurer* au sein du phage *Lactococcus Lactis p2.* Cette protéine possède bien le rôle qui lui est attribué puisque nous y avons trouvé une très belle unité de mesure dont la période est parfaite et présente une bonne conservation. Ce *ruban à mesurer* (ADC80089.1) présenté en Section 3.5.2, a une taille de 999 acides aminés (2997 nucléotides). Le génome hôte de cette protéine (GQ979703.1) a une taille de 27 595 nucléotides et nous pouvons souligner que le *ruban à mesurer* représente 10,86% de la taille totale du génome. Enfin, le NCBI propose l'outil *Graphical Sequence Viewer* afin de visualiser les différents gènes dans un génome. Dans le génome du phage *Lactococcus Lactis p2*, nous observons que

2. Voir section 1.2.3

le gène qui code la plus grosse protéine est bien celui du *ruban à mesurer* (Figure 3.2).

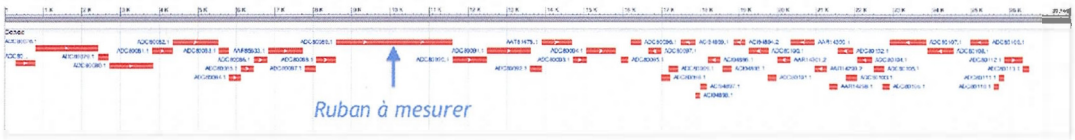


Figure 3.2: Représentation du génome du phage *Lactococcus Lactis p2* (GQ979703.1) dans le *Graphical Sequence Viewer*. En rouge sont représentés les gènes appartenant au génome du phage *Lactococcus Lactis p2*, en respectant leurs tailles et leurs positions. Le gène qui code pour la protéine *ruban à mesurer* (ADC80089.1) est assurément le plus long de tout le génome.

Afin de revenir à l'évaluation des annotations, nous retenons la règle suivante : si une protéine *ruban à mesurer*, issue d'un génome de bactériophage, est la plus grosse du génome alors l'annotation est validée sinon, elle est rejetée.

2. **La séquence provient d'un génome de bactérie donc, d'un prophage.** Un génome de bactérie hôte d'un prophage peut contenir plusieurs prophages, tel qu'expliqué en Section 1.5. Ainsi, il pourrait également y avoir plusieurs *rubans à mesurer*, de tailles différentes, au sein du même génome et nous ne pouvons pas être aussi restrictifs que précédemment. À l'aide de *rubans à mesurer* déjà identifiés, nous avons essayé différents seuil sur leurs génomes respectifs ; nous avons alors établi que 98% des plus grosses protéines du génome correspondaient à un seuil acceptable. Ainsi, si une protéine *ruban à mesurer*, issue d'un génome de bactérie, fait partie des 98% plus grosses protéines du génome, elle peut passer à la prochaine étape de validation. Dans le cas des prophages, nous possédons également une autre caractéristique permettant de valider l'annotation. Au sein des bactéries, les prophages sont traités comme des gènes intégrés à l'ADN génomique. L'étape d'intégration du génome du

bactériophage (le prophage) au sein de l'ADN de la bactérie est réalisée grâce à une protéine appelée *integrase*. Après intégration, l'*integrase* restera à l'intérieur du prophage, au sein du génome de la bactérie. Ainsi, nous devrions retrouver une *integrase* aux alentours du *ruban à mesurer* (Zhou et al., 2011). Nous utilisons la taille moyenne des génomes des bactériophages afin de clarifier les « *alentours* » du *ruban à mesurer* soit : ±50 000 nucléotides. Cette particularité est illustrée en Figure 3.3.

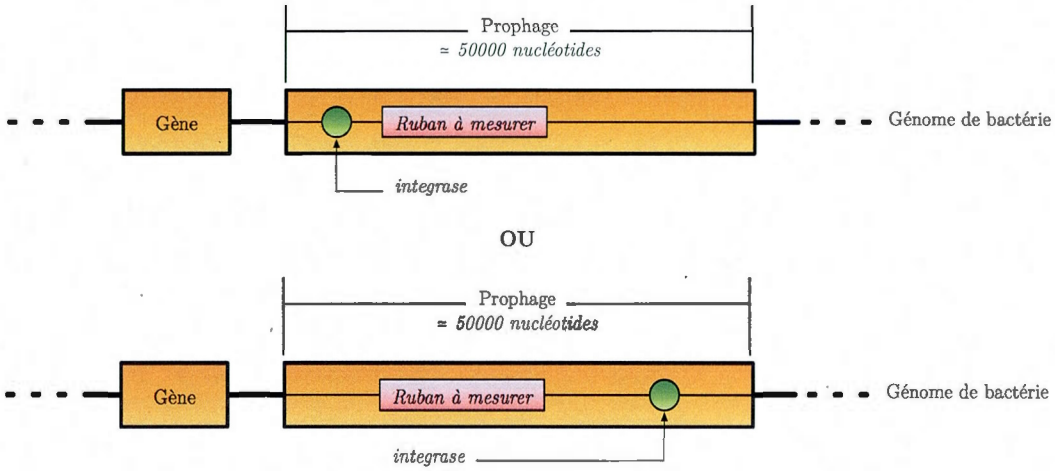


Figure 3.3: Localisation des protéines *integrase* dans les génomes de bactéries. Les génomes de bactéries comportent des gènes et des prophages qui sont traités comme des gènes. Les *integrases* se retrouvent au sein de prophages. Ainsi, à une distance de ±50 000 nucléotides d'un *ruban à mesurer*, nous devons retrouver une protéine *integrase*.

Un exemple de la présence de cette protéine *integrase* est proposé en Figure 3.4. Nous observons un *ruban à mesurer* issu d'un génome de bactérie *Enterococcus faecalis TX0630* (AEBE01000006.1) dans lequel nous avons identifié une unité de mesure à période constante, présentée en Section 3.5.4. À l'intérieur de l'éventail des ±50 000 nucléotides, nous avons retrouvé une *integrase*. Nous pouvons conclure que la protéine annotée *ru-*

ban à mesurer se situe au sein d'un prophage et donc, que l'annotation est correcte (si elle satisfait également la contrainte de la taille de la protéine. Sinon, l'annotation sera moins fiable, ou à réviser à la main).

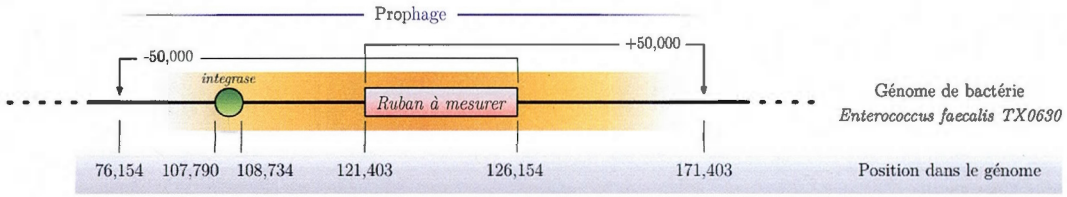


Figure 3.4: Exemple de localisation d'une protéine *integrase* dans un génome de bactérie *Enterococcus faecalis TX0630* (AEBE01000006.1). À l'aide des données du NCBI, nous connaissons la position du *ruban à mesurer* (EFU91923.1) dans le génome : 121,403..126,154. La contrainte que nous recherchons est la présence d'une protéine *integrase* afin de s'assurer que nous sommes au sein d'un prophage. En utilisant la taille moyenne d'un génome de bactériophage, nous recherchons une annotation *integrase* à ±50 000 nucléotides du *ruban à mesurer* : 76,154..171,403. Dans le génome de la bactérie *Enterococcus faecalis TX0630*, une protéine *integrase* est présente à la position 107,790..108,734 donc, à l'intérieur de l'éventail des ±50 000 nucléotides. Ainsi, dans ce cas, l'annotation *ruban à mesurer* satisfait la contrainte de présence d'une protéine *integrase*.

Le pseudo-code de notre algorithme de validation est présenté en Algorithme 1. Celui-ci a été programmé en Python en utilisant le module Biopython (Cock et al., 2009; Chapman et Chang, 2000).

Cet algorithme peut paraître assez simple, cependant lors de la programmation de celui-ci de nombreux paramètres bioinformatiques rentrent en compte. Nous disposons en entrée, de séquences *ruban à mesurer* en format FASTA. Ce format donne uniquement une séquence, sans aucune description à part l'annotation. Notre première étape consiste à identifier l'organisme. Pour cela, nous devons récupérer la fiche GenBank[3] qui corres-

3. Exemple de fichier GenBank en Appendice C.2

Algorithme 1 Validation des annotations *ruban à mesurer*

```
1: Procedure VALIDATION(fasta_file)          ▷ fasta_file : contient les séquences ruban à mesurer
2:     Pour chaque séquence dans faste_file Faire
3:         Si L'organisme est Virus (Bactériophage) Alors
4:             Si Le ruban à mesurer est la plus grosse protéine du génome Alors
5:                 Valider l'annotation
6:             Sinon
7:                 Rejeter l'annotation
8:             Fin Si
9:         Sinon (l'organisme est Bactérie ou Archéobactérie)
10:            Si Il y a une integrase à ±50000 nucléotides du ruban à mesurer Alors
11:                Si Le ruban à mesurer fait partie des 98% plus grosses protéines du génome Alors
12:                    Valider l'annotation                ▷ La séquence satisfait les 2 contraintes
13:                Sinon
14:                    Annotation moyennement fiable       ▷ ... satisfait uniquement la contrainte
   « integrase »
15:                Fin Si
16:            Sinon
17:                Si Le ruban à mesurer fait partie des 98% plus grosses protéines du génome Alors
18:                    Annotation moyennement fiable ▷ ... satisfait uniquement la contrainte « taille »
19:                Sinon
20:                    Rejeter l'annotation                ▷ La séquence ne satisfait aucune contrainte
21:                Fin Si
22:            Fin Si
23:        Fin Si
24:    Fin Pour
25: Fin Procedure
```

pond à notre séquence en entrée. Ce format est une fiche descriptive renseignant entre autres sur la taxonomie. Ensuite, nous avons besoin d'identifier les différentes protéines codées par le génome afin de s'assurer que le *ruban à mesurer* est bien la plus grosse. Pour cela, il faut récupérer le numéro d'accession du génome complet lié à la protéine ; celui-ci sera alors téléchargé au format GenBank afin de comparer les tailles des différentes protéines. Pour cela, nous parcourons la totalité du génome en enregistrant les tailles des protéines dans une liste. Nous avons également utilisé un filtre qui traite uniquement les génomes qui codent pour plus de 10 protéines cela afin d'exclure les génomes incomplets et d'éviter les faux positifs.

Dans le cas des génomes de bactéries et de la recherche de la présence d'une *integrase*, les positions dans le génome ainsi que les annotations *integrase* sont également effectuées sur un fichier GenBank. Les génomes des bactéries étant beaucoup plus gros, la récupération d'un génome complet est plus longue que dans le cas des génomes de virus. Enfin, tous les fichiers GenBank ne possèdent pas les mêmes caractéristiques et nous devons rechercher l'annotation *integrase* dans des champs qui ont des particularités différentes (*product* ou *misc_features* par exemple).

Un simple pseudo-code peut rapidement devenir complexe dans le cas des données et bases de données biologiques. Le code de ce programme Python est donné en Appendice A.3.

Après exécution du programme, nous avons écarté 971 séquences *ruban à mesurer* qui ne satisfaisaient aucune des contraintes. Nous avons désormais 1463 séquences dont 432

sont très fiables, satisfaisant toutes les contraintes et 1031 séquences légèrement moins fiables (elles sont issues de génomes de bactéries et ne satisfont qu'une seule des deux contraintes).

Un récapitulatif des traitements effectués sur le jeu de données initial est donné en Tableau 3.1.

Traitements	Séquences retenues	Séquences écartées
Requête initiale	6739	×
Suppression des doublons	2633	4106
Suppression des séquences dont la taille est inférieure à 200 acides aminés	2459	174
Suppression des séquences dont l'organisme est différent de *virus*, *bactérie* ou *archéobactérie*	2434	25
Exécution du programme d'évaluation de l'annotation	1463	971

Tableau 3.1: Tableau récapitulatif des traitements effectués sur notre jeu de données

Possédant un ensemble de données fiable et exploitable, nous allons rechercher des unités de mesure au sein des séquences *ruban à mesurer*. Ces recherches sont présentées dans la section suivante.

3.4 Recherche de duplications

Dans les séquences annotées *ruban à mesurer*, nous recherchons des duplications en tandem qui pourraient nous renseigner sur l'existence d'unités de mesure. Nous avons en premier lieu utilisé les outils bioinformatiques disponibles.

3.4.1 Outils bioinformatiques

Nous avons d'abord utilisé le programme de Gary Benson : *Tandem Repeats Finder* (présenté en Section 2.4.1). Cet outil est développé pour les duplications en tandem à frontières variables et il teste la totalité des possibilités de duplications.

Cependant, dans la plupart des séquences de protéines *ruban à mesurer*, il ne trouve aucune duplication. Elles ont subi trop de mutations durant l'évolution et aucune combinaison *frontière/période* ne présente de distance assez faible pour que le programme détecte une duplication. Comme vu précédemment en section 2.4.1, l'algorithme crée un consensus des sondes candidates et les aligne. Si la distance de cet alignement est faible, la duplication en tandem est validée. Dans le cas des protéines *ruban à mesurer*, cette distance ne sera jamais assez faible ainsi nous avons besoin d'utiliser d'autres méthodes afin d'identifier des unités de mesure.

Nous avons ensuite utilisé le programme de recherche de motifs MEME (présenté en Section 2.4.2) qui a retourné des résultats intéressants. En effet, dans la séquence donnée en Figure 3.1 celui-ci trouve un motif revenant à huit reprises. Ce motif est illustré en Figure 3.5 en format LOGO. Il a une période de 25 acides aminés et la hauteur des caractères représente leur fréquence d'apparition. Les caractères des colonnes 3 et 4 sont les plus hauts du motif et donc, les plus intéressants. Le tryptophane (`W`) est seul dans sa colonne et cela signifie qu'il est présent dans chacune des 8 répétitions du motif. La colonne de l'asparagine (`N`) est partagée avec un acide aspartique (`D`), plus petit. Cela indique que l'asparagine (`N`) est plus souvent présente à cette position (`N` : 7 fois

sur 8 ; `D` : une fois sur 8).

Figure 3.5: Motif trouvé par *MEME* dans la séquence de la Figure 3.1. Il a une longueur de 25 acides aminés. Les colonnes 3 et 4 sont les plus intéressantes. Le tryptophane (`W`) est seul dans la colonne 3 et cela signifie qu'il est conservé dans les 8 occurrences du motif. La colonne 4 est partagée entre une asparagine (`N`) et un acide aspartique (`D`). Le `N` est plus grand et est présent dans 7 occurrences du motif tandis que le `D` n'est présent qu'une fois.

Ce motif peut être aligné dans la séquence afin de mieux observer les duplications. Ainsi, en partant d'une séquence dans laquelle *Tandem Repeats Finder* ne trouvait aucune duplication, nous pouvons désormais mettre en évidence des séquences répétées de longueur fixe. Ces 8 motifs sont présentés en Figure 3.6 avec les `W` et `N` mis en évidence.

Ce résultat est particulièrement intéressant et permet de faire l'analogie avec les recherches de Siponen et al. (Siponen et al., 2009). Cette équipe de chercheurs a mis en évidence l'existence de deux parties globulaires aux extrémités du *ruban à mesurer*. Cela se traduit par l'existence de portions de séquences avant et après les duplications (les duplications correspondent à la partie « qui *mesure* »). Autrement dit, dans la séquence de la Figure 3.6, nous retrouvons les parties globulaires, en rouge, avant et après les huit répétitions.

Afin d'alléger le texte, les répétitions des *rubans à mesurer* seront désormais présentées sans leurs parties globulaires.

>gi|259503473|ref|ZP_05746375.1| phage tail tape measure protein
[Lactobacillus antri DSM 16041]

MAGSSLGHLAATVSLNINPFKASNNVLKAEIKSTANALRAQEIALKGSANSLNNMRAVYATMQSQMNNYN
AQLQRSKALMDDTTRSERSRLNAANQFNKTSAQVEILRTRMQALSREIVAQSSRWGQLAARANHFGNVAT
AVGSRVQGVGRGMSTYLTAPIAAGLAYSASKLVDFQDAMNRTKNVIRTSGESAAETQRSYNQMLRDSRKY
SDMYGVSQIKIANGYQDLVKRGYTSKAAVGVMRNELKASIATGDDFNDVIKVASQTMESFGLSTNKAGQP
LKSARLMQERSKDTLNKLAYAADATSTDFNSLGVGMSYVGATAHQAGFGLGETASAMGILSNNGLEADKA
GTGLRKVIQSLITPTKGGAEALAKINLSTKDFIGKNGKLKSMSTIFATLNKHMKGLTGHEKNDIFHSLFG
TTGQQAGAILTENAKRLRELTNEVNNAGKRDYIGDLAKRNLNTPKAQLAIFKESLTNAGMDAAKNVLPII
TPLVQDLSKLAQGFGQLPEPMKKAITYTTLFVAGAGPLLLLIGKLSSGVGGLALKLGSFAGGMSRARAAI
SMGAGGLTVLKSAFSKNAFETAKFGKAATTASGAMTTLKGTVDTANGTISSSKTIVDSAGRALTTAGESA
AVAGTSFGAAAIPIAAATAVIIAGVAAWELWGKKAAASDERTNRWGSDVGVAADHALSKMQSTSQGIQTA
LSDMDVAAHTSTKKMADDFDREFTQIERSAREHFKNVKKAEKGLSPEVSKAIDAEAEKEKSSFNGYVHDA
DVANQNVQKILSGRNRKVSELDQTQRTMLRNYQAEMLNDEMKTLQLSGSRRAKAMAVLNNDIKHMSQQQR
STRMGDITSETAEMEQQYKRQASLLKKRYKNNELSRAEYRAGLKSNQRALSDYSNKAAAEYIRLARANGE
STSLIKKDLKSMGLSYKAGTAEIRRQTKAAMQDQKSLAVNTTKMSGKVKKAA
NMWNNLVFDPKTGKVRTNAQEEVNKAVNSS
KGWNQIKLLKKEGKLSTNAQHMVAAALIAN
GQWDSMSWPDKQAWLHDKFHQTIVKALEDS
GQWNNLTLEQKEAIVNAKGKQEMADLLMES
VGWNNLTLQEKQAIVADKATLPLVDSLQKS
GQWNGMTLKQQEAIINAKGKDQLVDALVKG
GVWNSLSLKQQLALVRTQGTEQVIQGIDQM
GRWNSLSPKQQQAIVSAKGGPDLQQLITDY
GLWRGLPASAAKQIIAQDRASGNLKAANDAMQAWIKANPGAPKNALAIDNASGPLHNA
TGGVNVFAGSSTGAPKVASGVDNASGPFGNATNSIFGWNRTGANTKNIRANDQASKNAQIAISAIDAWNK
AHPIIHKFITEVITKHKNKANGANYFEGGPVMVNDQKGPIFREMVQFPGEIPFIPYGRNVLLNAPRGTRI
VKASDTLKQFPHLPQYANGNSDAVSVLNNIQPVQVSAKGTNDSQSASTSDNKYMGEVLERLGKMTDRLGT
MLGLNAAQLSAIKASAFDKNDLYSKMGMDQVYYDAQRL

Figure 3.6: Séquence de la Figure 3.1 dont le motif découvert par MEME a été mis en évidence. Les parties en rouge sont les parties globulaires du *ruban à mesurer*

Cependant, ces répétitions ne nous renseignent pas sur la présence d'une mesure ; nous avons juste mis en évidence le motif trouvé par *MEME*. D'autre part, la nature variable des séquences biologiques, qui peuvent accumuler des mutations tout en conservant la même fonction au sein de l'organisme, nous force à dépasser le cadre restreint d'une stricte répétition d'un seul motif.

3.4.2 Extension d'un motif

Le rôle particulier de l'acide aminé tryptophane (W) suggère de regarder toutes les occurrences de cet acide aminé dans la séquence.

La Figure 3.7 présente la séquence précédente (Figures 3.1 et 3.6) dans laquelle les tryptophanes (W) sont alignés en début de ligne, tout en gardant une période relativement régulière.

```
>gi|259503473|ref|ZP_05746375.1
phage tail tape measure protein
[Lactobacillus antri DSM 16041]

WNNLVFDPKTGKVRTNAQEEVNKAVNSSKG          (1)
WNQIKLLKKEGKLSTNAQHMVAAALIANGQ          (2)
WDSMSWPDKQAWLHDKFHQTIVKALEDSGQ          (3)
WNNLTLEQKEAIVNAKGKQEMADLLMESVG          (4)
WNNLTLQEKQAIVADKATLPLVDSLQKSGQ          (5)
WNGMTLKQQEAIINAKGKDQLVDALVKGGV          (6)
WNSLSLKQQLALVRTQGTEQVIQGIDQMGR          (7)
WNSLSPKQQQAIVSAKGGPDLQQLITDYGL          (8)
WRGLPASAAKQIIAQDRASGNLKAANDAMQA         (9)
WIKANPGAPKNALAIDNASGPLHNATGGVNV         (10)
FAGSSTGAPKVASGVDNASGPFGNATNSIFG         (11)
WNRTGANTKNIRANDQASKNAQIAISAIDA          (12)
WNKAHPIIHKFITEVITKHKNKANGANYFE          (13)
```

Figure 3.7: Séquence de la Figure 3.1 dont les W sont alignés en début de ligne

Des lignes (1) à (8), nous retrouvons les huit répétitions précédentes d'une période de 30 acides aminés. Elles ne sont plus liées à un motif, mais cadencées par l'acide aminé tryptophane (W). Des lignes (9) à (13), nous observons 5 duplications supplémentaires dont 3 dans lesquelles la période a légèrement varié passant de 30 à 31 acides aminés. Cette base supplémentaire aux duplications (9), (10) et (11) peut être due à une mu-

tation appelée *l'insertion*. La période de longueur 30 revient ensuite aux lignes (12) et (13).

À la ligne (11), la mesure n'est pas un alignement avec le tryptophane, mais avec la phénylalanine (`F`). Cette mutation, au sein des protéines *ruban à mesurer*, a déjà été rencontrée par Belcaid et al. (Belcaid, Bergeron et Poisson, 2011). L'explication biologique par Siponen et al. (Siponen et al., 2009) propose la phénylalanine (`F`) comme marqueur alternatif au tryptophane (`W`).

En utilisant le format *ProSite* défini précédemment, le motif qui revient 12 fois est : « Un tryptophane ou une phénylalanine, suivis de 29 à 30 acides aminés » soit, `[FW]-x(29,30)`. Nous pouvons également noter six occurrences du motif `W-x(2)-L`. Ce motif sera rencontré plus loin, à de nombreuses reprises.

3.5 Unités de mesure cadencées par le tryptophane (`W`)

Les premières démarches ayant mis en évidence le lien entre l'unité de mesure et le tryptophane, nous nous sommes concentrés sur cette relation. Nous avons donc écrit un programme Python qui aligne les tryptophanes (`W`) en début de ligne afin de faire apparaître des unités de mesure dans les séquences retenues. Certaines séquences n'ont donné aucun résultat tandis que d'autres ont révélé des motifs répétés très intéressants. Des exemples de piètre résultat et de résultat fructueux sont présentés en Figure 3.8 et Figure 3.9.

```
>gi|227885990|ref|ZP_04003795.1| tail tape measure protein,family
prophage MuMc02 [Escherichia coli 83972]

WALKYNQYQDELQEAVGSLISDNIDNVSDIGFLMPDIARAATATRTSAQD
WAKVAAV
WQNSLKGAARDFGAVQNIMAYAGDQGSFEIPDQVK
WMQSLAPMMAGLASGKEAVAEIGASLQIAKIGAGSTDEAANNFKNFLTKIFARDTQKQFADLGIDLQGSIAS...
WVATHPQFVSGAFKLISALLAIKIATIGLKLGLNLLISPFVNV
WKTTVLLRTN
WHRLTTALGEGGKLR
WLVTGFSRLTSGGLKLSKVLAGSLVRGFMSAARAVL
WIGRALMMNPIGLVITAVATAAYLIYRN
WGAVSG
WFKQR
WADIQEAFNGGIVGIGKLLIN
WSPAGLLYKAFAAALKYFGVALPAKFTDFGGHLIDGLINGIKTNGGRSNPV
```

Figure 3.8: Exemple de piètre résultat (ZP_04003795.1). On ne peut distinguer aucune unité de mesure dans cette séquence. L'alignement des tryptophanes (W) ne fait apparaître aucune période évidente, même en regroupant des lignes.

```
>gi|304439227|ref|ZP_07399145.1
conserved hypothetical protein
[Peptoniphilus duerdenii ATCC BAA-1640]

WKKCD
WFREGVISI
WDTIKESTVAI
WNGIKEFFVNL
WQGISDSWTST
WTEITSFLSEF
WSGFIEGVKTT
WKGIKDFFANL
WNGLSEGWNSI
WTSITTFLTES
WNTFIEGAKSL
WQSLGEFFTSL
WTGIQTTFTNI
WTAISTTTTEVFTAVGEFIKTT
WEGIKTLISTVLDAIKVKVETI
WNGLKEFLTTVITAIGEFISTS
WTNIKTTIETILTSIKTVLESV
WNGIKTFISSTMNNIKTFVSSA
WNSIKSTISSAVNSAKSAVSSA
FNSMRSSISSTMSNIQFTIRNG
FNNAVNHIKNLASQAYTWGADM
```

Figure 3.9: Exemple de résultat intéressant (ZP_07399145.1). Une période régulière de longueur 11 est apparue à la suite de l'alignement des tryptophanes (W). Cette répétition revient 11 fois et, on observe également 8 répétitions de longueur 22 (un multiple de 11) ainsi que 2 mutations du tryptophane (W) en phénylalanine (F).

3.5.1 Groupement par familles

Comme le *ruban à mesurer* est crucial lors de l'assemblage du bactériophage, nous espérons observer une grande conservation de l'unité de mesure, même lorsque des séquences deviennent distantes.

Pour cela, nous avons groupé les séquences par pourcentage d'identité avec le programme *CD-Hit* (Li et Godzik, 2006). Le pourcentage d'identité est une mesure de la

ressemblance entre deux séquences. Sur un alignement de séquences, le pourcentage d'identité est donc le pourcentage de bases identique, aux mêmes positions, dans les 2 séquences. Un exemple d'identité à 60% est présenté ci-dessous, où 6 des 10 acides aminés sont identiques. L'étoile sous une colonne indique l'identité parfaite.

```
WNNLVFDTKT
WKQLVFDLKY
*  ***  *
```

L'explication détaillée des principes de comparaison de séquences est donnée en Appendice B.

Nous avons convenu qu'une similarité supérieure à 70% crée des groupements intéressants. En dessous de 70%, les séquences deviennent trop distantes et nous retrouvons des motifs d'unités de mesure totalement différents. Les pourcentages d'identités dans les familles sont calculés par rapport à un *représentant*, qui est la séquence qui a le plus grand pourcentage d'identité avec toutes les autres. Un exemple de famille contenant 5 séquences, dont le pourcentage d'identité minimal par rapport au représentant est 71,38%, est présenté en Figure 3.10.

Après groupement, nous avons 914 familles dont 4 très grandes possédant plus de 15 séquences et 862 petites, de moins de 4 séquences. La taille moyenne d'une famille est comprise entre 1 et 2 séquences. Cette petite taille est due aux 720 familles ne possédant

```
>Cluster 52
0 1341aa, >gi|302131828|ref|ZP... at 79.19%
1 1344aa, >gi|330989571|gb|EGH... at 79.32%
2 1348aa, >gi|330962206|gb|EGH... *     <----
Représentant
3 1069aa, >gi|330940154|gb|EGH... at 71.38%
4  640aa, >gi|213967374|ref|ZP... at 71.41%
|    |            |                  |
1    2            3                  4
```

Figure 3.10: Exemple de groupement : La famille 52. La première colonne correspond aux numéros de séquence au sein de la famille. La deuxième est la longueur de la séquence en acides aminés. La troisième correspond aux numéros d'accession. Enfin, la quatrième et dernière colonne donne les pourcentages d'identité des séquences par rapport au *représentant* de la famille. La séquence 2 est le *représentant* de la famille, symbolisé par l'étoile « * ». Elle est la séquence qui a le plus grand pourcentage d'identité avec toutes les autres. Elle a une longueur de 1348 acides aminés et possède comme numéro d'accession EGH62466.1.

qu'une seule séquence (À l'opposé, la plus grande famille contient 41 séquences). Les 866 familles de tailles extrêmes ont été élaguées dans un premier temps et nous travaillons d'abord avec les 48 familles de taille comprise entre 4 et 15 séquences. Cela représente un total de 324 séquences.

Nous disposons donc d'une stratégie de recherche d'unités de mesure ainsi que d'un groupement des séquences en familles. Nous présentons, dans les sections suivantes, les familles ayant retourné les résultats les plus intéressants.

3.5.2 Famille 42 (*Lactococcus Lactis Phage p2*)

La famille 42 contient 6 séquences dont une appartenant au phage *Lactococcus Lactis p2* étudié par l'équipe de Marina Siponen (Siponen et al., 2009). La séquence de son

ruban à mesurer est le représentant de la famille (séquence 0), donnée en Figure 3.11. Les unités de mesure des 6 séquences de la famille sont présentées en Figure 3.13.

```
>Cluster 42
0 999aa, >gi|289063541|gb|ADC... *     <----     Représentant
1 996aa, >gi|255985805|gb|ACU... at 81.12%
2 999aa, >gi|255985700|gb|ACU... at 84.28%
3 996aa, >gi|255985644|gb|ACU... at 77.21%
4 996aa, >gi|255985591|gb|ACU... at 80.32%
5 999aa, >gi|115315588|ref|YP... at 78.38%
```

Cette famille présente un motif complexe, mais très intéressant. L'alignement des tryptophanes (`W`) a présenté une légère constance, mais il a fallu regrouper des lignes afin de voir apparaître 19 fois le motif `[WF]-x(10)`, dans lesquels sont intercalés 8 fois le motif `[WF]-x(17)`. Les périodes de longueur 18 aux lignes (3), (5), (10), (13), (17), (20), (23) et (26), surviennent aléatoirement et nuisent à la régularité du motif. De plus, le pourcentage d'identité moyen entre les duplications est très faible : 26%. Cependant, le pourcentage d'identité devient très élevé entre des duplications distantes : les lignes (19), (22) et (25) ont un pourcentage d'identité de 73% et sont séparées par exactement 29 acides aminés. Nous avons alors recherché une unité de mesure différente au sein de cette séquence.

À l'aide des pourcentages d'identité obtenus précédemment, nous avons identifié 9 fois le motif régulier `[WF]-x(39)`. Nous avons à nouveau pris en compte la mutation du tryptophane (`W`) en phénylalanine (`F`), survenue aux lignes (4), (8) et (9). Ce motif alternatif est présenté en Figure 3.12 dans laquelle les `W` et `F` ont été mis en évidence. En plus de posséder une période parfaite de 40 acides aminés, on retrouve aux colonnes

```
>gi|289063541|gb|ADC80089.1|
tail tape measure protein
[Lactococcus lactis phage p2]

FFTQTETGKKA          (1)
WASFVDFLKSA          (2)
WDGIVSFFSGIGQWFADI   (3)
WNGAVDGAKGI          (4)
WQGLVDWFSGIVQGVQNI   (5)
WNGITTFFTTL          (6)
WTTVVTGIQTA          (7)
WAGVTGFFTGL          (8)
WDGIVNVVTTV          (9)
FTTISSLVTGAYNWFVTT  (10)
FQPLISFYKSI         (11)
FGLVGSVINLA         (12)
FQLILAIIRGAYQLVIGA  (13)
WSGISGFFGVI         (14)
FNAVSSVVSTV         (15)
FSAIGSFAGSA         (16)
WNVLVGVWNAVAGFFGGI  (17)
FNAVKGVVSSV         (18)
FSAIGSFASSA         (19)
WGVVSSIWSAVSGFFSGI  (20)
FNAVSSVVSGV         (21)
FSALGGFASNA         (22)
WGAITGIFSGVADFFSGV  (23)
FDGAKNIVSGV         (24)
FEAFGNFASNA         (25)
WNAITGVFNGIGSFFSDI  (26)
FGGVKNTIDSV         (27)
```

Figure 3.11: Unité de mesure chez le phage *Lactococcus Lactis p2*. Représentant de la famille 42 (ADC80089.1). Les lignes (19), (22) et (25) ont un fort pourcentage d'identité (73%) et sont séparées par exactement 29 acides aminés.

7, 8 et 12 des alignements de **W** et **F** supplémentaires qui pourraient produire des unités de mesure alternatives.

```
>gi|289063541|gb|ADC80089.1| tail tape measure protein
[Lactococcus lactis phage p2]

WASFVDFLKSAWDGIVSFFSGIGQWFADIWNGAVDGAKGI          (1)
WQGLVDWFSGIVQGVQNIWNGITTFFTTLWTTVVTGIQTA          (2)
WAGVTGFFTGLWDGIVNVVTTVFTTISSLVTGAYNWFVTT          (3)
FQPLISFYKSIFGLVGSVINLAFQLILAIIRGAYQLVIGA          (4)
WSGISGFFGVIFNAVSSVVSTVFSAIGSFAGSAWNVLVGV          (5)
WNAVAGFFGGIFNAVKGVVSSVFSAIGSFASSAWGVVSSI          (6)
WSAVSGFFSGIFNAVSSVVSGVFSALGGFASNAWGAITGI          (7)
FSGVADFFSGVFDGAKNIVSGVFEAFGNFASNAWNAITGV          (8)
FNGIGSFFSDIFGGVKNTIDSVLGGVTDTINNIKGSIDWV          (9)
|     ||   |          |     |    |
12345678901234567890123456789012345678901
```

Figure 3.12: Unité de mesure alternative chez le phage *Lactococcus Lactis p2* (ADC80089.1). Elle est la plus longue unité de mesure qui présente les plus grands pourcentages d'identité entre duplications au sein des séquences de la famille 42

Grâce à cet exemple, nous pouvons insister sur le fait que différentes unités de mesure peuvent être mises en évidence au sein d'une même séquence. Cependant, afin de faire un choix parmi les différentes unités de mesure observables, nous avons utilisé les pourcentages d'identité entre les lignes. Ainsi, l'unité de mesure présentée en Figure 3.12 est la plus longue que l'on puisse trouver au sein de la séquence et, elle est celle qui retourne les plus grands pourcentages d'identité entre les duplications.

Le motif **[WF]-x(39)** est bien plus satisfaisant que **[WF]-x(10)**, car la grande conservation entre les duplications est en accord avec la théorie des duplications en tandem. Cette conservation est plus spécialement observable des lignes (5) à (8), qui présentent les plus grands pourcentages d'identité. Prenons par exemple, les lignes (6) et (7) de l'unité de

0 >gi|289063541|gb|ADC80089.1| tail tape measure protein [Lactococcus lactis phage p2] WASFVDFLKSAWDGIVSFFSGIGQWFADIWNGAVDGAKGI WQGLVDWFSGIVQGVQNIWNGITTFFTTLWTTVVTGIQTA WAGVTGFFTGLWDGIVNVVTTVFTTISSLVTGAYNWFVTT FQPLISFYKSIFGLVGSVINLAFQLILAIIRGAYQLVIGA WSGISGFFGVIFNAVSSVVSTVFSAIGSFAGSAWNVLVGV WNAVAGFFGGIFNAVKGVVSSVFSAIGSFASSAWGVVSSI WSAVSGFFSGIFNAVSSVVSGVFSALGGFASNAWGAITGI FSGVADFFSGVFDGAKNIVSGVFEAFGNFASNAWNAITGV FNGIGSFFSDIFGGVKNTIDSVLGGVTDTINNIKGSIDWV	1 >gi|255985805|gb|ACU46987.1| putative tape measure protein [Lactococcus phage CB20] WASFVDFLKSAWDGIVTFFSGIGQWFSDIWNGAVDGAKGI WQGLVDWFSGVVQGIQNIWNGIKGFFTTLWTTVVTGIQTA WAGVTGFFTGLWDGIVNIVTTVFTTIATLVTGAYNWFVKT FQPLISFYQSIFGLIGSIINLAFQLILAVIRGAYKSVLVA WQGLSAWFSGIFNVVKSVVSIAFSAIGSFASSAWNLVRSV WSEITGFFGGIFNYVRGIVSGSFSAIGSFASSAWGVVSSI WSAASGFFSGIFNSVRSVVSGVFNSLGGFASNAWRAISDA FSGVGNFFSGTFNGAKSTVSGVFSSLGGFASNAYKAITDA FSGIGDFFSRIFGGVKSTIDNALGGVTNTINHISGAINGI
2 >gi|255985700|gb|ACU46884.1| putative tape measure protein [Lactococcus phage CB14] WASFVDFLKSSWDGIVTFFSGMGQWFSDIWNGAVDGAKGI WQGLVDWFSGVVQGIQNIWNGIKGFFTTLWTTVVTGIQTA WAGVTGFFTGLWDGIVNIVTTVFTTIATLVTGAYNWFVKT FQPLISFYQSIFGLIGSIINLAFQLILAVIRGAYKSVLVA WQGLSAWFSGIFNVVKSVVSIAFSAIGSFASSAWNLVRSV WSEITGFFGGIFNYVRGIVSGSFSAIGSFASSAWGVVSSI WSAASGFFSGIFNSVRSVVSGVFNSLGGFASNAWRAISDA FSGVGNFFSGTFNGAKSTVSGVFSSLGGFASNAYKAITDA FSGIGDFFSRIFGGVKSTIDNALGGVTNTINHISGAINGI	3 >gi|255985644|gb|ACU46829.1| putative tape measure protein [Lactococcus phage CB13] WASFVDFLKSAWDGIVSFFSGMGQWFSDIWNGAVDGAKGI WQGLVDWFSGVVQGIQNIWNGIKGFFTTLWTTVVTGIQTA WAGVTGFFTGLWDGIVNVVKTAFTTIASLVTGAYNWFVTT FQPLISFYQSIFGLIGSIINLAFQLILAVIRGAYELVINA WQGLSAWFSGIFNVVKSVVSTAFSAIGSFASNAWNLVRSV WSAISGFFSSIFNTVKSVVSSVFSALGGFASNAWNAVKGV FSSVGSWFGGVFDSAKKAVSDALGTLGGFASNAWNAIKSA FSSVGSWFSGVFDSAKKAVSDALGTLGNIAKGAWDSIKNA FGGVHDFFAKAFGGVKDLVDNALGGISGTIDKISGAINGV
4 >gi|255985591|gb|ACU46778.1| tape measure protein [Lactococcus phage SL4] WASFVDFLKSAWDGIVSFFSGMGQWFSDIWNGAVDGAKGI WQGLVDWFSGVVQGIQNIWNGIKGFFTTLWTTVVTGIQTA WAGVTGFFTGLWDGIVNIVKTVFTTIATLVTGAYNWFVKT FQPLISFYQSIFGLIGSIINLAFQLILAVIRGAYKSVLVA WQGLSAWFSGIFNVVKSVVSIAFSAIGSFASSAWNLVRSV WSEITGFFGGIFNYVRGIVSGAFSAIGSFASSAWGVVSSI WSAASGFFSGIFNSVRSVVSGVFNSLGGFASNAWRAISDA FNGVGNFFSGAFNGVKSTVSGVFSSLGGFASNAYKAITDA FSGIGDFFSRIFGGVKSTIDNALGGVTNTINHISGAINGI	5 >gi|115315588|ref|YP_764275.1| tail tape measure [Lactococcus phage 712] WASFVDFLKSAWDGIVSFFSGIGQWFADIWNGAVDGAKGI WQGLVDWFSGIVQGIQNIWNGITTFFTTLWTTVVTGIQTA WTGVTEFFTGLWNGIVTIVTTVFTTIASLVTGAYNWFVTT FQPLISFYQSIFNLIGSIINLAFQLILAIIRGAYQLVVNA WKGISGFFGVIFNAVKSVVSTVFSAIGSFAGSAWNVLVGV WNAVSGFFGGIFNAVKGVVSSVFSAIGSFASSAWGVVSSI WNAVSGFFSRIFNTVKSVVSSAFSALGGFASNAWNAITGV FSSVGSWFGGVFDSAKEAVSNALGALGNIAKGAWDSITSV FGGVYDFFANAFGGVKDLIDNILGGISGTLDKISGAINGV

Figure 3.13: Unités de mesure dans les séquences de la famille 42

mesure du phage *p2* (séquence 0 : ADC80089.1), qui est la répétition présentant le plus grand pourcentage d'identité. Un alignement de ces deux lignes nous a retourné un pourcentage d'identité de 67,5% avec une différence de seulement 13 acides aminés. Cette forte identité entre les lignes (6) et (7) se retrouve dans toutes les séquences de la famille. En reprenant les bases des duplications en tandem, la duplication qui a la plus grande identité correspondrait à la plus récente duplication (soit, les lignes (6) et (7)). Les alignements de ces lignes sont présentés en Figure 3.14 pour les séquences 0 et 5 ; les lignes (6) et (7) de la séquence 5 (YP_764275.1) ont un pourcentage d'identité est de 65%.

```
seq0_ligne_six          WNAVAGFFGGIFNAVKGVVSSVFSAIGSFASSAWGVVSSI
seq0_ligne_sept         WSAVSGFFSGIFNAVSSVVSGVFSALGGFASNAWGAITGI
                        * ** *** ******  *** **** * *** ***    *

seq5_ligne_six          WNAVSGFFGGIFNAVKGVVSSVFSAIGSFASSAWGVVSSI
seq5_ligne_sept         WNAVSGFFSRIFNTVKSVVSSAFSALGGFASNAWNAITGV
                        ********  *** ** **** *** * *** **
```

Figure 3.14: Alignement des répétitions ayant les plus grands pourcentages d'identité dans les séquences 0 (ADC80089.1) et 5 (YP_764275.1). Les étoiles sous les colonnes correspondent à une identité parfaite. On observe bien une très grande similarité entre les séquences et, si on fait le lien avec les acides aminés *intéressants*, on observe qu'aucun tryptophane (W) ou phénylalanine (F) n'est perdu.

3.5.3 Famille 36

La famille 36 est composée de 6 séquences.

```
>Cluster 36
0 1617aa, >gi|229551679|ref|ZP... at 82.75%
1 1624aa, >gi|227535666|ref|ZP... *     <----
Représentant
2 1621aa, >gi|22217809|emb|CAD... at 73.60%
3 1620aa, >gi|195661213|ref|YP... at 72.10%
4 1621aa, >gi|258539075|ref|YP... at 74.40%
5 1620aa, >gi|199598974|ref|ZP... at 72.22%
```

L'alignement des tryptophanes dans ces six séquences a mis en évidence des répétitions de période quasi fixe, illustrées en Figure 3.15. Cette famille est particulièrement intéressante parce que les séquences qu'elle contient sont assez distantes (un minimum de 72,10% et un maximum de 82,75% d'identité par rapport au représentant), mais les unités de mesure sont tout de même conservées. Cette particularité montre que l'unité de mesure ne se perd pas au cours de l'évolution.

Ainsi, au sein de cette famille, nous avons découvert le motif **W-x(29,30)**, répété 11 fois. Cette répétition est illustrée sur le représentant de la famille (séquence 1 : ZP_03965715.1), en Figure 3.16. De plus, nous avons émis l'hypothèse qu'un lien existe entre l'unité de mesure et la leucine (L). En effet, on observe au début de chaque répétition, le motif W-x(2)-L.

Ce motif d'unité de mesure est conservé dans les séquences 0, 1, 3 et 5 de la famille. Cependant, dans les séquences 2 et 4 le motif diffère légèrement en présentant une nouvelle fois, la mutation d'un tryptophane en phénylalanine (W en F). La leucine (L)

0 >gi|229551679|ref|ZP_04440404.1| TP901 family phage tail tape measure protein [Lactobacillus rhamnosus LMS2-1] WNNLVFDTKTGKVKTNLPEVLKDTASTKKG WKQLVFDLKYAKITSNAKQMIVEALASSEQ WQKLSVPEKNAIIRTQGREQLADIMDKFVS WNSLSLKDQQVIVKGDYTPLVNALVKSGD WNNLTLKQQEAIVKDKATVPLVSSLQQTGE WQKLDLKVQEAIVNAKGKKDLEDIIFDMGV WNKLPNTQKYATLVSFGKQDIADIIDQLNL WNTLTPKEIKAVAKGDTSSLIAAIDKAND WNRLTLGQLEAIVKDKASAGLVQAMIKAGE WNGLSIEEKTAIMQTKGKSDLADMVVKYGL WNSLPNSTKSLLMNDSDARTKLEKAGVAID	1 >gi|227535666|ref|ZP_03965715.1| TP901 family phage tail tape measure protein [Lactobacillus paracasei ATCC 25302] WNSLVFDPKTGKVKTNLPEVLKDTASTKKG WQQLKFDLKNAKITSNAKQMIVEALASSKQ WQKLSVPEKNAIIRTQGREQLADIMDKFVS WNSLSLKDQQAIVKGDYTPLVNALVKSGD WNNLTLKQQEAIVKDKATAPLVSSLQQTGE WQKLDLKVQEAIVNAKGKKDLEDILFDMGV WNKLPNTQKYATLVSFGKQDIADIIDQLNL WNTLTPKEIQAVAKGDTSSLVAAIDKAND WNRLTLGQLEAIVKDKASAGLVQAMIKTGE WNGLSIEEKTAIMQTKGKSDLADMVVKYGL WNSLPNSTKSLLMNDSDARTKLEKAGVAID
2 >gi|22217809|emb|CAD43903.1| tape measure protein [Lactobacillus phage A2] WNKLVLDPKTGKVITNLPEVLHDTANTKEG WKRLTFDLKNAKISTNAKETIAIALASTDK WNSLSVDEKNAIIKETGRKDLADLMKRMVS WNDLTLEQQQAVVKGDYAPLIDAIIQAGT WNQLDVEDKQVLVKDKANIPLVDALVNSGR WNKLDLKTQNALLQAKGKKDLEDVLFNMGL WNSLDMNEKYAQLKAIGKTDLADMIDQLGL WDTITPKQMEAAVKGDYSQLTAAIDQVHG WNQLDTKQLEAIVQDKATVPLIQAMIQNQK WNGLSVEEKNAILKTKGMPELADMVVKYGS FDSLPDSTKRLLINDDDARQKLIAAGVNMD	3 >gi|195661213|ref|YP_002117681.1| tape measure protein [Lactobacillus phage Lrm1] WNSLVLDPKTGKVVTNLPQVLKDTASTEGG WKRLKFDLKNAKISSNAKQMIVEAMASTDK WNSLTVQEKTALVRASGKKELANIITEFVS WNKFTPKEQQAIVSGDYTPLVNALVKMGY WNELSLKEQQAIVHDKATLPLIDILTQSGK WQGLTLKQQTALINAKGKDELKDVLFNLGV WQSLDPKDQYTTLKAVGEGKLADMLDQLAL WNKITPQQMQAVVKGDYSSLVTAIDEVNG WNQLTPKQMQMIVQDKATATLIQGMIEAQS WNRLSVEAKTALIQAKGKEQLADAVAKFGL WNQLPSKTKELLVNNADARAKLVEAGIDVD
4 >gi|258539075|ref|YP_003173574.1| hypothetical protein LC705_00884 [Lactobacillus rhamnosus Lc 705] WNKLVLDPKTGKVITNLPEVLHDTANTKEG WKRLTFDLKNAKISTNAKETIAIALASTDK WNSLSVDEKNAIIKETGRKDLADLMKRMVS WNDLTLEQQQAVVKGDYAPLVDAIIQAGT WNQLDVEDKQVLVKDKANIPLVDALVNSGR WNKLDLKTQNALLQAKGKKDLEDVLFNMGL WNSLDMNEKYAQLKAIGKTDLADMIDQLGL WDTITPKQMEAAVKGDYSQLTAAIDQVHG WNQLDTKQLEAIVQDKATVPLIQAMIQNQK WNGLSVEEKNAILKTKGMPELADMVVKYGS FDSLPDSTKRLLINDDDARQKLIAAGVNMD	5 >gi|199598974|ref|ZP_03212383.1| hypothetical protein LRH_06956 [Lactobacillus rhamnosus HN001] WNNLVLDPKTGKVVTNLPQVLKDTASTEDG WKRLKFDLKNAKISSNAKQMIVEAMASTDK WNSLTVQEKTALVRASGQKDLANIITEFVS WNKFTPKEQQAIVSGDYTPLVNALVQMGY WNELSLKEQQAIVHDKATLPLIDILTQSGK WQGLTLKQQTALINAKGKDELKDVLFNLGV WQSIDPKDQYTTLKAVGEGKLADMLDQLAL WNKITPQQMQAVVKGDYSSLVTAIDEVNG WNQLTPKQMQMIVQDKATATLIQGMIEAQS WNRLSVEAKTALIQAKGKEQLADAVAKFGL WNQLPSKTKELLVNNADARAKLVEAGIDVD

Figure 3.15: Unités de mesure dans les séquences de la famille 36

étant conservée, on retrouve alors en ligne (11) le motif **F-x(2)-L**. Cet exemple de mutation est présenté en Figure 3.17 sur la séquence 4.

```
>gi|227535666|ref|ZP_03965715.1|
TP901 family phage tail tape measure protein
[Lactobacillus paracasei ATCC 25302]

WNSLVFDPKTGKVKTNLPEVLKDTASTKKG          (1)
WQQLKFDLKNAKITSNAKQMIVEALASSKQ          (2)
WQKLSVPEKNAIIRTQGREQLADIMDKFVS          (3)
WNSLSLKDQQAIVKGDYTPLVNALVKSGD           (4)
WNNLTLKQQEAIVKDKATAPLVSSLQQTGE          (5)
WQKLDLKVQEAIVNAKGKKDLEDILFDMGV          (6)
WNKLPNTQKYATLVSFGKQDIADIIDQLNL          (7)
WNTLTPKEIQAVAKGDTSSLVAAIDKAND           (8)
WNRLTLGQLEAIVKDKASAGLVQAMIKTGE          (9)
WNGLSIEEKTAIMQTKGKSDLADMVVKYGL         (10)
WNSLPNSTKSLLMNDSDARTKLEKAGVAID         (11)
```

Figure 3.16: Représentant de la famille 36 (ZP_03965715.1)

```
>gi|258539075|ref|YP_003173574.1|
hypothetical protein LC705_00884
[Lactobacillus rhamnosus Lc 705]

WNKLVLDPKTGKVITNLPEVLHDTANTKEG          (1)
WKRLTFDLKNAKISTNAKETIAIALASTDK          (2)
WNSLSVDEKNAIIKETGRKDLADLMKRMVS          (3)
WNDLTLEQQQAVVKGDYAPLVDAIIQAGT           (4)
WNQLDVEDKQVLVKDKANIPLVDALVNSGR          (5)
WNKLDLKTQNALLQAKGKKDLEDVLFNMGL          (6)
WNSLDMNEKYAQLKAIGKTDLADMIDQLGL          (7)
WDTITPKQMEAAVKGDYSQLTAAIDQVHG           (8)
WNQLDTKQLEAIVQDKATVPLIQAMIQNQK          (9)
WNGLSVEEKNAILKTKGMPELADMVVKYGS         (10)
FDSLPDSTKRLLINDDDARQKLIAAGVNMD         (11)
```

Figure 3.17: Séquence 4 de la famille 36 (YP_003173574.1). La ligne (11) a subi une mutation **W** en **F**.

3.5.4 Famille 17

Cette famille est composée de 10 séquences appartenant à des phages qui infectent l'espèce de bactérie *Enterococcus* dont la séquence 1 est le représentant. Tout comme la famille 36, certaines séquences sont assez distantes (avec un pourcentage d'identité minimum de 76,95% par rapport au représentant).

```
>Cluster 17
0 1582aa, >gi|315580351|gb|EFU... at 78.26%
1 1583aa, >gi|315579732|gb|EFU... *     <----
Représentant
2 1582aa, >gi|315170940|gb|EFU... at 79.71%
3 1583aa, >gi|315162187|gb|EFU... at 96.21%
4 1582aa, >gi|315033058|gb|EFT... at 78.32%
5 1583aa, >gi|307271315|ref|ZP... at 98.29%
6 1553aa, >gi|312902460|ref|ZP... at 76.95%
7 1567aa, >gi|257421733|ref|ZP... at 78.43%
8 1568aa, >gi|257081795|ref|ZP... at 95.15%
9 1582aa, >gi|257078057|ref|ZP... at 82.05%
```

Les unités de mesure de ces 10 séquences sont données en Figure 3.18. Le motif `W-x(29,30)` revient de 8 à 9 reprises dans toutes les séquences de cette famille.

Ici, la période des répétitions ne varie pas aléatoirement : toutes les séquences (hormis la 6) possèdent 4 répétitions `W-x(29)` suivies de 5 fois `W-x(30)`. Ce motif d'unité de mesure est donné en Figure 3.19 avec la séquence du représentant du groupe (séquence 1 : EFU91923.1). On retrouve également le motif `W-x(2)-L` à 6 reprises sur les 9 répétitions.

0

>gi|315580351|gb|EFU92542.1| phage tail tape measure protein, TP901 family, core region [Enterococcus faecalis TX0309A]
WNDLVLEDKDGHVKSNAQEAITEAAKSEQG
WNKLLFASKKAELKSNAKLMIAEAAIANGR
WEKMTFTEQQALLDSNVTKTMTQALQAKGS
WGKLNFEEKKAVLYSNTPEVMAETMLNLGL
WKDYQPQIKELKAKNQSFLEVLSQSQDKIVH
WSQVPVDIKEILGDNYDLLSKIYSSEQSYNR
WKNLPDDEKKLLANNSDVLQKIMTSDTSLKQ
WNALPAEQKKMLGDNTDLMSKVFASKESFDM
WSKIPDNVKRMLANNEDLQNKLMQGHLSISD

1

>gi|315579732|gb|EFU91923.1| phage tail tape measure protein, TP901 family, core region [Enterococcus faecalis TX0630]
WNNLVLDPKTGEVKTNAQEAVNEAANSEKG
WNQLLYASKHADLKSNAKLMIAEAAIANGK
WDSMTFKEQQALLDTNAKKTVTQALQANGK
WDKLNFEEKKAILYSNTPEKMAENMLNLGL
WEDYKLHDKEIKADNKEFLEVLSDSQEKIVN
WSNIPDDVKEFYADNQDLLTKIYGSERAFNA
WKNLPDESKLLLANNTDVLQKILSSETYLTN
WNNLPTDQKKMLANNDDLLTKVMKSEESMNA
WKSLPDPVKKMLGNNEDLKAKIADGTLSVQT

2

>gi|315170940|gb|EFU14957.1| phage tail tape measure protein, TP901 family, core region [Enterococcus faecalis TX1342]
WNQLILDTKTGEVKTNAQEAVNEAASSEKG
WNQLIYASKNADLKSNAKLMIAEAAIANGR
WEKMTFTEQQALLDSNVTKTMTQALQAKGS
WGKLNFEEKKAVLYSNTPEVMAETMLNLGL
WKDYQPQVKELKAKNQSFLDVLSQSQDKIVH
WSQVPVDIKEILGDNYDLLSKIYGSEQSYNR
WKNLPDTEKKLLANNSDVLQKIMTSDTSLKQ
WNALPADQKKMLGDNTDLLTKVMASEESFNA
WKALPDPVKKMLGNNEDLKAKIAEGSLSVET

3

>gi|315162187|gb|EFU06204.1| phage tail tape measure protein, TP901 family, core region [Enterococcus faecalis TX0645]
WNNLVLDPKTGEVKTNAQEAVNEAANSEKG
WNQLLYASKHADLKSNAKLMIAEAAIANGK
WDSMTFKEQQALLDTNAKKTVTQALQANGK
WDKLNFEEKKAILYSNTPEKMAENMLNLGL
WEDYKLHDKEIKADNKEFLEVLSDSQEKIVW
WSNIPDDVKEFYADNQDLLTKIYGSERAFNA
WKNLPDESKLLLANNTDVLQKILSSETYLTN
WNNLPTDQKKMLANNDDLLTKVMKSEESMNA
WNSLPDPVKKMLGNNEDLKAKIADGTLSVQT

4

>gi|315033058|gb|EFT44990.1| phage tail tape measure protein, TP901 family, core region [Enterococcus faecalis TX0017]
WNQLILDTKTGEVKTNAQEAVNEAASSEKG
WNQLIYASKNADLKSNAKLMIAEAAIANGR
WEKMTFTEQQALLDSNVTKTMTQALQAKGS
WGKLNFEEKKAVLYSNTPEVMAETMLNLGL
WKDYQPQVKELKAKNQSFLDVLSQSQDKIVH
WSQVPVDIKEILGDNYDLLSKIYGSEQSYNR
WKNLPDDEKKLLANNSDVLQKIMTSDTSLKQ
WNALPADQKKMLGDNTDLLTKVMASEESFNA
WKLLPDPVKKMLGNNEDLEAKIADGTISIEE

5

>gi|307271315|ref|ZP_07552594.1| phage tail tape measure protein, TP901 family, core region [Enterococcus faecalis TX0855]
WNNLVLDPKTGEVKTNAQEAVNEAANSEKG
WNQLLYASKHADLKSNAKLMIAEAAIANGK
WDSMTFKEQQALLDTNAKKTVTQALQANGK
WDKLNFEEKKAILYSNTPEKMAENMLNLGL
WEDYKLHDKEIKADNKEFLEVLSDSQEKIVN
WSNIPDDVKEFYADNQDLLTKIYGSERAFNA
WKNLPDESKLLLANNTDVLQKILSSETYLTN
WNNLPTDQKKMLANNDDLLTKVMKSEESMNA
WNSLPDPVKKMLGNNEDLKAKIADGTLSVQT

6

>gi|312902460|ref|ZP_07761666.1| phage tail tape measure protein, TP901 family, core region [Enterococcus faecalis TX0635]
WNNLVLDPKTGEVKTNAQEAVNEAANSEKG
WNQLLYASKHADLKSNAKLMIAEAAIANGK
WDKLNFEEKKAILYSNTPEKAENMLNLGL
WEDYKLHDKEIKADNKEFLEVLSDSQEKIVN
WSNIPDDVKEFYADNQDLLTKIYGSERAFNA
WKNLPDESKLLLANNTDVLQKILSSETYLTN
WNNLPTDQKKMLANNDDLLTKVMKSEESMNA
WKSLPDPVKKMLGNNEDLKAKIADGTLSVQT

7

>gi|257421733|ref|ZP_05598723.1| predicted protein [Enterococcus faecalis X98]
WNQLILDTKTGEVKTNAQEAVNEAASSEKG
WNQLIYASKNADLKSNAKLMIAEAAIANGR
WEKMTFTEQQALLDSNVTKTMTQALQAKGS
WGKLNFEEKKAVLYSNTPEVMAETMLNLGL
WKDYQPQVKELKAKNQSFLDVLSQSQDKIVH
WSQVPVDIKEILGDNYDLLSKIYGSEQSYNR
WKNLPDDEKKLLANNSDVLQKIMTSDTSLKQ
WNALPADQKKMLGDNTDLLTKVMASEESFNA
WKLLPNPVKKMLGNNEDLEAKIADGTISIEE

8

>gi|257081795|ref|ZP_05576156.1| reticulocyte binding protein [Enterococcus faecalis E1Sol]
WNNLVLDPKTGEVKTNAQEAVNEAANSEKG
WNQLLYASKHADLKSNAKLMIAEAAIANGK
WDSMTFKEQQALLDTNAKKTVTQALQANGK
WDKLNFEEKKAILYSNTPEKMAENMLNLGL
WEDYKLHDKEIKADNKEFLEVLSDSQEKIVN
WSNIPDDVKEFYADNQDLLTKIYGSERAFNA
WKNLPDESKLLLANNTDVLQKILSSETYLTN
WNNLPTDQKKMLANNDDLLTKVMKSEESMNA
WNSLPNSVKKMLGNNEDLKAKIADGTLSVQT

9

>gi|257078057|ref|ZP_05572418.1| conserved hypothetical protein [Enterococcus faecalis JH1]
WNDLVLEDKDGHVKSNAQEAITEAAKSEQG
WNKLLFASKNADLKSNAKLMIAEAAIANGR
WEKMTFTEQQALLDSNVTKTMTQALQAKGS
WGKLNFEEKKAVLYSNTPEVMAETMLNLGL
WKDYQPQVKELKAKNQSFLDVLSQSQDKIVH
WSQVPVDIKEILGDNYDLLSKIYGSEQSYNR
WKNLPDDEKKLLANNSDVLQKIMTSDTSLKQ
WNALPADQKKMLGDNTDLLTKVMASEESFNA
WKLLPDPVKKMLGNNEDLKAKIADGTLSVQT

Figure 3.18: Unités de mesure dans les séquences de la famille 17

```
>gi|315579732|gb|EFU91923.1|
phage tail tape measure protein, TP901 family,
core region [Enterococcus faecalis TX0630]

WNNLVLDPKTGEVKTNAQEAVNEAANSEKG          (1)
WNQLLYASKHADLKSNAKLMIAEAAIANGK          (2)
WDSMTFKEQQALLDTNAKKTVTQALQANGK          (3)
WDKLNFEEKKAILYSNTPEKMAENMLNLGL          (4)
WEDYKLHDKEIKADNKEFLEVLSDSQEKIVN         (5)
WSNIPDDVKEFYADNQDLLTKIYGSERAFNA         (6)
WKNLPDESKLLLANNTDVLQKILSSETYLTN          (7)
WNNLPTDQKKMLANNDDLLTKVMKSEESMNA         (8)
WKSLPDPVKKMLGNNEDLKAKIADGTLSVQT         (9)
```

Figure 3.19: Unité de mesure du représentant de la famille 17 (EFU91923.1)

Un membre particulièrement remarquable de cette famille est la séquence 6 (ZP_07761666.1) où le motif **W-x(29)** est répété 3 fois au lieu de 4. Cette différence avec les autres séquences nous mène à nous interroger sur cette mutation. Elle peut être une disparition d'une répétition au complet dans la séquence 6, ou bien le gain d'une répétition dans toutes les autres séquences. En tout état de cause, cette mutation survenue sur une répétition au complet est une preuve supplémentaire du rôle de mesure que possèdent les séquences répétées au sein des *rubans à mesurer*.

Une mutation biologique, connue sous le nom de *délétion*, consiste en la perte de bases dans une séquence [4]. Nous avons vu que la taille du *ruban à mesurer* est liée à la taille de la queue du bactériophage. Ainsi, supprimer une unité dans le *ruban à mesurer* revient a réduire la taille de la queue du bactériophage. Parallèlement, *l'insertion* se définit

4. Voir Appendice B.2

comme le gain de bases et, l'ajout d'une unité rallongerait la queue du bactériophage.

Nous pouvons donc suggérer que la séquence 6 appartient à un phage dont la queue est plus courte d'une unité de mesure que les autres phages de la famille. L'unité de mesure présente dans ce bactériophage est donnée en Figure 3.20

```
>gi|312902460|ref|ZP_07761666.1|
phage tail tape measure protein, TP901 family,
core region [Enterococcus faecalis TX0635]

WNNLVLDPKTGEVKTNAQEAVNEAANSEKG          (1)
WNQLLYASKHADLKSNAKLMIAEAAIANGK          (2)
WDKLNFEEKKAILYSNTPEKKAENMLNLGL          (3)
WEDYKLHDKEIKADNKEFLEVLSDSQEKIVN         (4)
WSNIPDDVKEFYADNQDLLTKIYGSERAFNA         (5)
WKNLPDESKLLLANNTDVLQKILSSETYLTN         (6)
WNNLPTDQKKMLANNDDLLTKVMKSEESMNA         (7)
WKSLPDPVKKMLGNNEDLKAKIADGTLSVQT         (8)
```

Figure 3.20: Unité de mesure de la séquence 6, famille 17 (ZP_07761666.1). Une répétition entière est manquante entre les lignes (1) et (3), car nous retrouvons le motif `W-x(29)` uniquement trois fois. Dans les autres séquences de cette famille, nous retrouvons ce motif quatre fois, entre les lignes (1) et (4).

Afin de savoir où la mutation est survenue, nous avons aligné l'unité de mesure de la séquence 6 avec celle du représentant de la famille. Le résultat présenté en Figure 3.21, nous montre que c'est la répétition (3) qui a muté : soit par une *insertion* à la ligne (3a), soit par une *délétion* à la ligne (3b). Nous pouvons ajouter que cette répétition est la seule qui ne présente pas le motif `W-x(2)-L`. Dans les alignements ligne à ligne, nous pouvons observer une très grande identité entre les séquences où seul un acide aminé diffère sur les trois répétitions (mis en évidence en bleu).

```
>Unité de mesure séquence 1                    >Unité de mesure séquence 6
WNNLVLDPKTGEVKTNAQEAVNEAANSEKG      (1a)       WNNLVLDPKTGEVKTNAQEAVNEAANSEKG
(1b)
WNQLLYASKHADLKSNAKLMIAEAAIANGK      (2a)       WNQLLYASKHADLKSNAKLMIAEAAIANGK
(2b)
WDSMTFKEQQALLDTNAKKTVTQALQANGK      (3a)       ------------------------------
(3b)
WDKLNFEEKKAILYSNTPEKMAENMLNLGL      (4a)       WDKLNFEEKKAILYSNTPEKKAENMLNLGL
(4b)
```

```
>Séquence1      WNNLVLDPKTGEVKTNAQEAVNEAANSEKG      (1a)
>Séquence6      WNNLVLDPKTGEVKTNAQEAVNEAANSEKG      (1b)

>Séquence1      WNQLLYASKHADLKSNAKLMIAEAAIANGK      (2a)
>Séquence6      WNQLLYASKHADLKSNAKLMIAEAAIANGK      (2b)

>Séquence1      WDKLNFEEKKAILYSNTPEKMAENMLNLGL      (4a)
>Séquence6      WDKLNFEEKKAILYSNTPEKKAENMLNLGL      (4b)
```

Figure 3.21: Alignement des unités de mesure des séquences 1 et 6 (EFU91923.1 et ZP_07761666.1). La troisième répétition de l'unité de mesure a été soit perdue dans la séquence 6 soit, gagnée dans la séquence 0. Dans les alignements ligne à ligne, nous observons une très grande identité où seul un acide aminé diffère, à la ligne 4, sur les trois répétitions.

3.6 Motifs d'unités de mesure identifiés

Nous avons décrit ci-dessus trois familles qui présentent les particularités les plus intéressantes. Cependant, nous avons identifié un total de 9 motifs différents, sur un ensemble de 129 séquences groupées en 18 familles. Un récapitulatif des motifs identifiés est présenté en Tableau 3.2. Attention, certaines séquences peuvent contenir plusieurs motifs.

Nous avons précédemment réduit notre jeu de données en élaguant les familles qui possèdent moins de 4 séquences ou plus de 15 mais, elles n'ont pas pour autant été

Motifs	Nombre de familles qui contiennent le motif	Nombre de séquences qui contiennent le motif
[WF]-x(10)	9 (Familles 12, 14, 28, 42, 44, 62, 66, 81, 84 et 89)	63
[WF]-x(21)	1 (Famille 81)	3
[WF]-x(28)	1 (Famille 36)	6
[WF]-x(29)	2 (Familles 17 et 36)	16
[WF]-x(30)	1 (Famille 17)	10
[WF]-x(32)	1 (Famille 84)	4
[WF]-x(39)	1 (Famille 42)	6
[WF]-x(3)-K	4 (Familles 13, 15, 19, 58, 87 et 92)	36
[WF]-x(2)-L	1 (Famille 36)	6

Tableau 3.2: Tableau récapitulatif des motifs identifiés

abandonnées. Nous sommes allés rechercher les motifs d'unités de mesure identifiés au sein des séquences mises à l'écart. Nous avons également effectué ces recherches auprès des bases de données du NCBI. Cette étape d'enrichissement des familles est discutée dans le chapitre suivant.

CHAPITRE IV

ENRICHISSEMENT DES FAMILLES

L'*enrichissement* consiste à ajouter de nouvelles séquences. Dans ce chapitre, nous expliquons comment nous avons enrichi notre jeu de données. Nous avons commencé par récupérer les séquences contenant des unités de mesure dans celles mises à l'écart de notre jeu de données initial. Nous avons ensuite recherché de nouvelles séquences auprès des bases de données du NCBI en prenant soin de chercher des séquences qui ne sont pas encore annotées *ruban à mesurer*. Après avoir enrichi nos familles, nous avons étudié les longueurs des duplications en fonction de l'espèce de la bactérie qui est attaquée. Cela nous a alors permis de comparer la longueur des *rubans à mesurer* à la taille de la paroi cellulaire de la bactérie qui est attaquée par le phage.

4.1 Enrichissement par recherche de motifs

Le jeu de données contenant les séquences *ruban à mesurer* dont l'annotation est validée contient 1463 séquences (Tableau 3.1). Après groupement par familles CD-Hit, nous nous sommes concentrés sur l'étude de 324 séquences qui sont dans 48 familles de taille 4 à 15, laissant 1139 séquences orphelines (comprises dans 866 familles).

Nous avons précédemment identifié 9 motifs d'unité de mesure dans un total de 129 séquences groupées en 18 familles (Tableau 3.2). Nous avons pu rechercher ces motifs dans les 866 familles mises à l'écart, à l'aide d'expressions régulières. Cette étape a pour but de trier ces familles afin de récupérer celles qui contiennent des unités de mesure.

Auparavant, nous avions commencé par chercher des motifs uniquement cadencés par le tryptophane (`W`), mais nous avons observé que la phénylalanine (`F`) peut être un marqueur alternatif. Ainsi, lors de nos recherches, nous prenons en compte cette caractéristique. Nous avons écrit un programme au langage Python qui recherche les 9 unités de mesure dans les séquences orphelines.

Un exemple est proposé avec le motif `[WF]-x(10)`, répété 10 fois : nous possédions déjà 63 séquences comportant ce motif et après exécution du programme, nous avons pu en ajouter 13 supplémentaires.

En tout, cette étape nous a permis d'ajouter 82 séquences contenant des unités de mesure identifiées. Ces séquences avaient été groupées dans 70 familles par CD-Hit : 11 familles possèdent 2 à 3 séquences et 59 contiennent une séquence unique.

Au total, notre fichier contenant les unités de mesure identifiées compte désormais 211 séquences, groupées en 88 familles.

Prenons comme exemple, la famille 70 qui contient 3 séquences :

```
>Cluster 70
0 938aa, >gi|294614761|ref|ZP... *
1 938aa, >gi|294622504|ref|ZP... at 95.52%
2 929aa, >gi|314995388|ref|ZP... at 96.88%
```

Celle-ci avait été mise à l'écart car elle contient moins de 4 séquences. Nous pouvons désormais l'ajouter à notre jeu de données d'unités de mesure validées. Nous retrouvons au sein de ses 3 séquences, le motif **[WF]-x(10)**, à 14 reprises ainsi que **W-x(3)-K** de 12 à 13 reprises. Ces unités de mesure sont données en Figure 4.1.

0	1	2
>gi|294614761|ref| ZP_06694661.1| tape measure protein, putative [Enterococcus faecium E1636]	>gi|294622504|ref| ZP_06701526.1| tape measure protein, putative [Enterococcus faecium U0317]	>gi|314995388|ref| ZP_07860491.1| tape measure domain protein [Enterococcus faecium TX0133a01]
WEETKNNMIAV	WEETKNNMIAV	WEETKNNMIGV
WNNIKESAINI	WNNIKESAINI	WNNIKESALNI
WESIKNIFVSY	WESIKNIFVSY	WESIKNIFVSY
FTNIYFAALNI	FTNIYFAALNI	ITNIYFAALNI
WTGFKLTLINI	WTGFKLTLINI	WTGFKLTLINI
WNEVVSQAKSI	WNEVVNQAKSI	WNEVVSQAKSI
WINVKYFFINL	WLNIKYFFINL	WINVKYFFINL
WIDIKYFAIQK	WIDIKYFAIQK	WIDIKYFAIQK
WIELKFGIIQT	WIELKFSIIQT	WIELKFGIIQT
WIDLKYNAITT	WIDLKYNAITT	WIDLKYNAITT
WNNIKQFFKDT	WNNIKQFFKDT	WNNIKQFFKDT
WQNIKDIAYNT	WKNIKDTAYNT	WQNIKDIAYNT
WISIKNSMINT	WISIKNTMINT	WISIKNSMINT
WNNIKESFWNI	WNNIKDSFWNI	WNNIKESFWNI

Figure 4.1: Unités de mesure au sein des séquences de la famille 70. Cette famille avait été mise à l'écart car elle contient moins de 4 séquences. Après enrichissement du jeu de données, nous l'ajoutons car nous avons identifié le motif **[WF]-x(10)**, 14 fois.

Afin d'enrichir les familles, nous allons nous tourner vers les bases de données du NCBI.

Nous y soumettons nos séquences qui possèdent des unités de mesure identifiées afin d'obtenir des résultats qui ne sont pas annotés *ruban à mesurer*. Enfin, nous pourrons utiliser notre programme d'évaluation de l'annotation [1] afin de s'assurer que ces séquences possèdent bien les caractéristiques des *rubans à mesurer*.

Cette méthode nous permet d'enrichir la totalité des familles, qu'elles possèdent peu ou beaucoup de séquences.

Pour cela, nous utilisons l'heuristique d'alignement local : BLAST (Basic Local Alignment Search Tool) (Altschul et al., 1990), présenté dans la section suivante.

4.2 Heuristique d'alignement local : BLAST

BLAST est un programme d'alignement local de nucléotides et d'acides aminés. C'est une heuristique plus rapide que les algorithmes d'alignement optimums (comme Smith-Waterman par exemple). Un alignement local aligne une séquence sur une autre, à l'endroit où la similarité est la plus élevée. La *similarité* est la mesure de la ressemblance entre deux séquences.

BLAST permet d'interroger directement les bases de données du NCBI. Ainsi, en soumettant un fragment de séquence, on peut savoir de quelles séquences il se rapproche le plus. Les résultats sont classés selon différents paramètres, mais dans notre cas, nous utilisons uniquement la *couverture de la séquence soumise* et le *pourcentage d'identité*.

La *couverture de la séquence soumise* est exprimée sous forme de pourcentage. Elle

1. Décrit en Section 3.3

donne le pourcentage de séquence soumise qui a été aligné. Par exemple, une séquence soumise longue de 10 acides aminés qui s'est alignée avec une séquence retournée longue de 5 acides aminés donnera une *couverture* de 50% (5 acides aminés / 10). Cet exemple est illustré en Figure 4.2

Comme vu précédemment, le *pourcentage d'identité* est le pourcentage de bases identique, aux mêmes positions, dans l'alignement des 2 séquences. Dans le premier exemple d'alignements donné en Figure 4.2, les 5 acides aminés alignés sont identiques, ce qui donne un pourcentage d'identité de 100%. Dans le second exemple, seuls 3 acides aminés sur 5 sont identiques, le pourcentage d'identité est alors 60%.

```
                          1234567890
Séquence soumise (Query)  WLGDKIGDVW
                             |||||          Couverture : 50% (5/10)
Séquence retournée (Subject)    DKIGD       Pourcentage d'identité : 100% (5/5)
                                12345
```

```
                          1234567890
Séquence soumise (Query)  WLGDKIGDVW
                             |||||          Couverture : 50% (5/10)
Séquence retournée (Subject)    DKFLD       Pourcentage d'identité : 60% (3/5)
                                12345
```

Figure 4.2: Alignements BLAST : Couverture et pourcentage d'identité

4.3 Évaluation du nombre de *rubans à mesurer* non annotés

Étant donné le grand nombre de phages dans les banques de données[2], nous pensons que la plupart de leurs séquences ne sont pas encore annotées, incluant les *rubans à mesurer*. Nous avons donc écrit un programme qui effectue les recherches BLAST sur les représentants des 88 familles créées précédemment. Nous avons uniquement soumis les répétitions au sein des séquences (les parties *qui mesurent*) et supprimé les parties globulaires[3].

Nous avons limité la *couverture minimale de séquence soumise* à 50%, le *pourcentage d'identité minimum* à 70% (en accord avec nos familles) ainsi qu'un nombre de résultats maximum de 500 séquences. Dans ces résultats, nous avons extrait les séquences qui ne sont pas annotées *ruban à mesurer* et vérifié qu'elles contiennent bien une unité de mesure à l'aide des expressions régulières.

Après exécution, notre jeu de données a été enrichi de 384 séquences non annotées *ruban à mesurer*.

Afin de valider les séquences ajoutées dans nos familles, nous utilisons notre programme d'évaluation de l'annotation, décrit en Section 3.3. Nous avons volontairement choisi des séquences qui ne sont pas annotées *ruban à mesurer* et cela a nécessité de trier nos résultats. En effet, parmi nos 384 séquences certaines sont en cours d'évaluation par le

---

2. Voir Section 1.3

3. Voir Figure 3.6 pour rappel

NCBI. Elles ne sont présentement reliées à aucun génome (numéros d'accession WP, par exemple) et ne peuvent pas être évaluées par notre programme.

Après avoir mis à l'écart les séquences ne pouvant pas être évaluées, nous conservons 56 séquences que nous allons soumettre à notre programme d'évaluation de l'annotation.

Sur 56 séquences, 11 ont été rejetées : 5 ne répondaient pas aux critères *taille* et/ou *integrase* et les 6 autres provenaient d'organismes non admissibles (différents de bactérie, archéobactérie ou virus). Ainsi, 45 séquences validées sur 56 représentent un taux de réussite de 80% à l'enrichissement du jeu de données par nos programmes.

En soumettant les représentants des 88 familles, nous avons ajouté 45 séquences non annotées *ruban à mesurer* à notre jeu de données ; il contient désormais 256 séquences. Les nouvelles séquences contiennent une unité de mesure identifiée : elles ont ensuite été ajoutées aux 88 familles existantes.

4.4 Enrichissement des familles avec les séquences non annotées *ruban à mesurer*

Notre programme traite les séquences représentantes des familles une à une. Pour chacune d'elles, il retourne les séquences qui viendront enrichir la famille. Nous ajoutons alors ces nouvelles séquences à celles déjà présentes dans la famille.

Sur les 88 familles, 34 ont reçu de nouvelles séquences. La famille 2 a été agrémentée de 17 nouvelles séquences : elle est la famille qui a le plus été enrichie.

Un exemple est présenté en Figure 4.3 dans lequel la famille 16 qui contenait 6 séquences a été enrichie de 3 séquences supplémentaires.

```
>Cluster 19
0 1617aa, >gi|229551679|ref|ZP... at 82.75%
1 1624aa, >gi|227535666|ref|ZP... *
2 1621aa, >gi|22217809|emb|CAD... at 73.60%
3 1620aa, >gi|195661213|ref|YP... at 72.10%
4 1621aa, >gi|258539075|ref|YP... at 74.40%
5 1620aa, >gi|199598974|ref|ZP... at 72.22%

- Avant Enrichissement -
```

```
>Cluster 19
0 1617aa, >gi|229551679|ref|ZP... at 82.75%
1 1624aa, >gi|227535666|ref|ZP... *
2 1621aa, >gi|22217809|emb|CAD... at 73.60%
3 1620aa, >gi|195661213|ref|YP... at 72.10%
4 1621aa, >gi|258539075|ref|YP... at 74.40%
5 1620aa, >gi|199598974|ref|ZP... at 72.22%
6 912aa,  >gi|524755654|emb|CD... >70%
7 1567aa, >gi|327409409|ref|YP... >70%
8 1593aa, >gi|525227496|emb|CD... >70%

- Après Enrichissement -
```

Figure 4.3: Exemple d'enrichissement de la famille 16

Nous devons insister sur le fait que, désormais les séquences au sein des familles peuvent avoir des tailles qui varient beaucoup plus qu'auparavant (tel qu'on peut l'observer en Figure 4.3). L'utilitaire CD-Hit utilisé pour créer les familles calcule les pourcentages d'identité entre des séquences alignées sur leur ensemble (*globalement*). Tandis que BLAST calcule ce même pourcentage *localement*, à l'endroit où s'est alignée la séquence. Ainsi, les pourcentages d'identité des alignements retournés par BLAST et de CD-Hit sont différents d'où la notation « `>70%` ».

Dans nos requêtes BLAST, nous avons configuré un pourcentage d'identité minimal de 70% entre la séquence soumise et les séquences retournées, puis retenu uniquement les séquences dans lesquelles nous avons identifié une unité de mesure. Cela nous permet de nous assurer que nous avons respecté le pourcentage d'identité désiré sur un motif d'unité de mesure et que notre enrichissement est correct.

4.5 Partage d'unités de mesure entre différentes espèces de phages

Nous avons précédemment enrichi les familles à l'aide de séquences non annotées *ruban à mesurer*. Ces séquences ont été obtenues par des recherches BLAST qui ont retourné des résultats variés. Parmi ces résultats, nous nous intéressons aux séquences qui proviennent d'espèces différentes de celle de la requête. Par exemple, si nous soumettons à BLAST une séquence issue d'une bactérie d'espèce *Listeria*, nous allons chercher les séquences provenant d'autres espèces. Cette étude permettra d'observer si des unités de mesure peuvent être partagées entre différentes espèces de phages et de bactéries.

Nous avons alors recherché les familles qui possèdent des séquences provenant de plusieurs espèces différentes à l'aide d'un programme Python. Sur 88 familles, 37 possèdent des séquences provenant d'au moins 2 espèces différentes. La famille 22 qui contient 17 séquences présentant le motif `[WF]-x(21)`, possède 5 espèces différentes. Elle est la famille la plus diversifiée.

La famille 25 possédait, avant enrichissement, 2 séquences issues d'une bactérie *Lactobacillus*. Celle-ci a été enrichie de 7 séquences supplémentaires et compte désormais 3 espèces différentes, elle est présentée en Figure 4.4. Toutes les séquences possèdent le motif `[WF]-x(29)`.

```
>Cluster 25
0 1500aa, >gi|338204462|ref|YP... at 91.33%          (Lactobacillus)
1 1503aa, >gi|194466413|ref|ZP... *                  (Lactobacillus)
2 1400aa, >gi|525868361|ref|YP... >70%               (Listeria)
3 1596aa, >gi|525919656|ref|YP... >70%               (Listeria)
4 1599aa, >gi|157325429|ref|YP... >70%
(Virus-Siphoviridae)
5 1599aa, >gi|525868561|ref|YP... >70%               (Listeria)
6 1600aa, >gi|525899440|ref|YP... >70%               (Listeria)
7 1599aa, >gi|525927611|ref|YP... >70%               (Listeria)
8 1599aa, >gi|525927215|ref|YP... >70%               (Listeria)
```

Figure 4.4: Famille 25, contient 9 séquences provenant de 2 espèces de bactérie différentes et d'un virus. Les séquences en bleu proviennent de l'enrichissement de la famille à l'aide de BLAST. Elles ne sont donc pas annotées *ruban à mesurer*. La famille initiale comportait deux séquences issues d'une espèce de bactérie *Lactobacillus*. Désormais, les séquences de cette famille proviennent de 3 espèces différentes : *Lactobacillus*, *Listeria* et le virus *Siphoviridae*.

Afin de conclure notre étude des *rubans à mesurer*, nous avons étudié la longueur moyenne des *rubans à mesurer* (en nombre de répétitions) selon l'espèce de bactérie, afin d'observer s'il existe un lien avec l'épaisseur de la membrane cellulaire.

4.6 Lien entre la longueur du *ruban à mesurer* et l'épaisseur de la membrane cellulaire

Nous avons vu précédemment que la taille du *ruban à mesurer* est liée à la taille de la queue du bactériophage[4]. Nous avons alors calculé les nombres moyens de répétitions par espèce de phage afin de les classer par taille. Les résultats des 6 plus grands *rubans à mesurer* sont donnés en Tableau 4.1.

Espèce de la bactérie attaquée par le phage	Nombre moyen de répétitions dans le *ruban à mesurer*
Bacillus	14
Lactobacillus	10
Staphylococcus	9
Streptococcus	9
Enterococcus	8
Clostridium	7

Tableau 4.1: Taille moyenne des unités de mesure (en nombre de répétitions) dans les séquences *ruban à mesurer* en fonction de l'espèce de la bactérie attaquée par le phage

Lors de l'étape d'infection, le bactériophage utilise sa queue afin de transpercer la membrane cellulaire de la bactérie. Nous avons alors cherché s'il existe un lien entre la longueur des *rubans à mesurer* et l'épaisseur de la membrane cellulaire de la bactérie

4. Voir Section 1.6 pour plus de détails

qui est attaquée. Ces données ont été trouvées dans des sources biologiques et les résultats sont présentés en Tableau 4.2.

Espèce de la bactérie attaquée par le phage	Nombre moyen de répétitions dans le *ruban à mesurer*	Taille de la membrane cellulaire
Bacillus	14	$22nm \pm 2$ (Beveridge et Graham, 1991)
Lactobacillus	10	10 à $20nm$ (Beveridge et Graham, 1991)
Staphylococcus	9	$18nm$ (Beveridge et Graham, 1991; Wyatt, 1970)
Streptococcus	9	$16nm \pm 1$ (Zuber et al., 2006)
Enterococcus	8	$18nm \pm 2$ (Zuber et al., 2006)
Clostridium	7	3 à $6nm$ (Beveridge et Graham, 1991)

Tableau 4.2: Taille moyenne des unités de mesure (en nombre de répétitions) dans les séquences *ruban à mesurer* et épaisseur des membranes cellulaires en fonction de l'espèce de la bactérie attaquée par le phage

Les résultats sont très intéressants puisque la tendance nous montre que plus la paroi cellulaire de la bactérie est épaisse, plus les phages qui l'attaquent auront un long *ruban à mesurer*. Le virion du bactériophage n'est qu'un moyen de transport pour l'ADN viral afin de se déplacer pour infecter une bactérie [5] et celui-ci devra nécessairement s'adapter a son hôte sans quoi, il ne pourra pas se répliquer.

5. Tel qu'expliqué en Section 1.5

Afin d'illustrer ce lien, nous avons tracé un graphique, présenté en Figure 4.5. Celui-ci permet d'observer la longueur des *rubans à mesurer* en fonction de la taille de la paroi cellulaire de la bactérie qui est attaquée. La courbe de tendance valide l'hypothèse que plus la paroi cellulaire d'une bactérie est épaisse, plus le *ruban à mesurer* sera long.

Ce lien biologique fort intéressant conclut notre étude des *rubans à mesurer.*

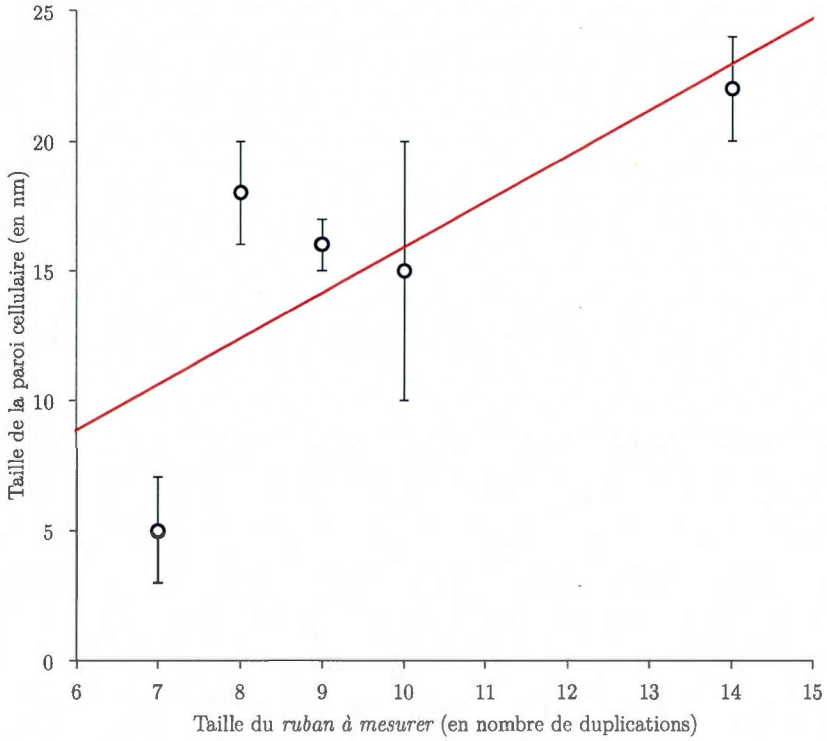


Figure 4.5: Nuage de points et courbe de tendance représentant la longueur des rubans à mesurer en fonction de la taille de la paroi cellulaire de la bactérie attaquée. Les points représentent les tailles moyennes des parois cellulaires et les barres d'erreurs correspondent aux tailles données dans les sources biologiques. La courbe de tendance démontre bien que plus la paroi cellulaire d'une bactérie sera épaisse, plus la taille du *ruban à mesurer* sera grande.

CONCLUSION

Après avoir filtré convenablement notre jeu de données, nous avons créé des groupements de séquences avec un pourcentage d'identité minimal de 70%. Dans ces familles, nous avons recherché des duplications en nous basant sur l'hypothèse que le tryptophane (W) rythme l'unité de mesure. Ces recherches à l'aide d'un programme Python et de validations à la main ont permis d'identifier 9 motifs d'unités de mesure dans 129 séquences ainsi qu'une mutation importante à prendre en compte : la phénylalanine (F) peut être un marqueur alternatif au tryptophane.

Nous avons alors mis en place un programme qui recherche les expressions régulières de ces motifs au sein de notre jeu de données afin de n'écarter aucune séquence qui contient une unité de mesure. Ayant conservé 211 séquences, groupées en 88 familles, nous avons effectué un enrichissement de ces familles avec des recherches BLAST sur les représentants. Après cela, notre jeu de données composé de séquences comportant des unités de mesure identifiées contient 256 séquences annotées et non annotées *ruban à mesurer*. Cet ensemble de données a été trié afin d'observer la longueur moyenne des répétitions par espèce de bactériophage. Cette étude a permis de mettre en valeur le lien entre la longueur du ruban à mesurer et l'épaisseur de la paroi cellulaire de la bactérie.

Bien que très intéressante, cette étude pourrait être poussée plus loin. Comme nous possédons plusieurs points de repère concernant les *rubans à mesurer*, il serait intéressant

de rechercher les différents motifs dans les bases de données du NCBI : à l'aide d'expressions régulière, contrairement aux recherches BLAST sur les unités de mesure identifiées. De plus, les bactériophages sont très nombreux dans les océans nous pourrions alors rechercher des *rubans à mesurer* dans les bases de données environnementales.

Ayant participé à l'étude du programme de reconstruction de l'histoire des duplications en tandem de Philippe Lavoie-Mongrain, il serait également intéressant de soumettre nos unités de mesure afin de valider leur reconstruction (surtout dans le cas des multiples de 11, cela permettrait de vérifier quel multiple est la meilleure reconstruction).

Enfin, nous pourrions contribuer à différents projets afin d'identifier des souches virales méconnues. Nous possédons un ensemble de données nous permettant d'identifier des *rubans à mesurer* et donc, des bactériophages dans n'importe quelle séquence. Par exemple, Cantalupo et al. ont séquencé des prélèvements faits dans les égouts de Pittsburgh, Barcelone et Addis Abeba afin d'identifier un grand nombre de sources virales (Cantalupo et al., 2011). Notre expérience nous permettrait de contribuer à l'identification de certains bactériophages.

APPENDICE A

CRÉATION DE LA BASE DE DONNÉES DE SÉQUENCES D'ACIDES AMINÉS ANNOTÉES *RUBAN À MESURER*

A.1 Récupération des séquences annotées *ruban à mesurer*

Afin de récupérer toutes les séquences d'acides aminés annotées *ruban à mesurer*, on utilise l'interface de programmation du NCBI : les utilitaires *Entrez*. Plus particulièrement, la fonction de recherche *eSearch* ainsi que la fonction de formatage des séquences *eFetch*. La requête *eSearch* est enregistrée au NCBI et retourne une clé permettant de récupérer les résultats dans le format qui nous convient à l'aide de *eFetch*.

Requête *eSearch* :

```
http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi
?db=protein&term=tape+measure&usehistory=y
```

En entrée : La base de données dans laquelle récupérer les séquences (**protein**), les termes de la recherche (**tape+measure**, pour *ruban à mesurer*) et l'utilisation de l'historique pour *eFetch* (**y** pour *yes*).

En sortie : Un fichier XML informe du nombre de résultats et donne les clés liées à

l'historique de cette recherche (**QueryKey** et **WebEnv**), présentée ci-dessous.

```
...
<QueryKey>1</QueryKey>
<WebEnv>
NCID_1_6772297_130.14.22.33_5555_1369166225_544897588
</WebEnv>
...
```

Grâce à ces clés stockées au NCBI, la fonction *eFetch* sait quelles séquences elle doit récupérer. Cette fonction propose également différents paramètres de formatage des séquences. Pour notre cas, nous voulons un fichier au format FASTA.

Requête *eFetch* :

```
http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=protein&query_key=1&WebEnv=
NCID_1_6772297_130.14.22.33_5555_1369166225_544897588&rettype=fasta&retmode=text
```

En entrée : La base de données à interroger (**protein**), les clés de la précédente recherche *eSearch* (**query_key=** et **WebEnv=**) et le formatage des résultats (format **fasta**). Le paramètre **retmode** est utilisé pour d'autres types de recherches au NCBI, comme les publications scientifiques par exemple. Dans notre cas, nous indiquons juste **text**.

En sortie : Téléchargement du fichier FASTA contenant toutes les séquences annotées *ruban à mesurer*. Notons que *eFetch* ne peut télécharger plus de 10000 séquences dans un fichier ainsi, les recherches plus grandes doivent être récupérées en plusieurs fois. Pour cela, il faut utiliser le paramètre **retstart** (**retstart=10001** afin de récupérer la suite d'une recherche).

Cette requête est très facilement programmable en Python à l'aide de Biopython et du module *Entrez*.

A.2 Suppression des doublons

Nous avons récupéré la totalité des séquences d'acides aminés annotées *ruban à mesurer* disponibles au NCBI, y compris celles en double. Pour supprimer les doublons, nous avons utilisé l'utilitaire CD-Hit [1] (Li et Godzik, 2006) qui a pour fonction de grouper des séquences par pourcentage d'identité. L'identité est le pourcentage de ressemblance lors de l'alignement de deux séquences [2].

Le pourcentage d'identité est un paramètre de CD-Hit. Dans notre cas, nous voulons supprimer les doublons et donc toutes les séquences identiques à 100%. Nous avons créé un jeu de données sans doublons avec 99% comme paramètre d'identité cela afin de supprimer les doublons ainsi que les séquences grandement similaires.

Lorsque toutes les séquences d'un groupement sont égales, c'est-à-dire lorsque l'identité est à 100%, CD-Hit en choisira une au hasard. Cependant, lorsqu'elles ont des pourcentages d'identité différents, il choisira celle qui a la plus grande identité avec les autres et elle alors sera le *représentant* du groupe (séquence 1 dans l'exemple ci-dessous, symbolisé par l'étoile « * »).

---

1. http ://cd-hit.org/

2. Pour plus de détails, voir Appendice B

```
>Cluster 490
0 1126aa, >gi|218890327|ref|YP_002439191.1... at 99.02%
1 1185aa, >gi|215480016|ref|YP_002332467.1... *
2 1126aa, >gi|218770550|emb|CAW26315.1..... at 99.02%
3 1185aa, >gi|169409280|gb|ACA57685.1..... at 100.00%
```

A.3 Évaluation de l'annotation

L'évaluation de l'annotation repose sur des caractéristiques propres aux phages et aux bactéries, expliquées en Section 3.3.

Le code au langage Python est donné ci-dessous.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-

from Bio import SeqIO
from Bio import Entrez
import time

Entrez.email = 'xx@x.org'
list_organism = []
count=0
virus_valides = open('valides_virus','w')
bacterie_valides = open('valides_bacterie','w')
fasta_tmp = open('tmp_valides.fasta', 'w')
annot_doute = open('tmp_douteuses.fasta', 'w')
annot_rejete = open('tmp_rejete.fasta', 'w')

for record in SeqIO.parse('fasta_more200_onlyBV.fasta','fasta'): #Pour ↙
  chaque sequence dans le fichier Fasta

  count = count+1

  print '\n','\n','####'

  print count,'>> --- Sequence TMP : ',record.id.split('|')[3] #Affiche le ↙
  GID de chaque sequence lue

  gid = record.id.split('|')[3]

  query = Entrez.efetch(db='protein', id=gid, rettype='gbwithparts', ↙
  retmode='text') #Requete au NCBI pour la sequence de la proteine ruban a ↙
  mesurer, en acides amines

```

```
29  query_file = open('query_file.gb','w')
30
31  for line in query:
32
33    query_file.write(line) #Ecrit le fichier Genbank
34
35
36  query_file = open('query_file.gb','r')
37
38  for rec in SeqIO.parse(query_file,'genbank'): #Biopython decode le fichier Genbank
39
40    print '--- ', rec.annotations['taxonomy'][0] #Affiche l'organisme de la sequence
41
42    if (rec.annotations['taxonomy'][0] == 'Viruses') : #Si l'organisme est VIRUS
43
44      for f in rec.features:
45
46        if f.type == 'CDS':
47
48          coded_by = f.qualifiers['coded_by'][0].split('complement(')
49
50          gid_coded_by = coded_by[len(coded_by)-1].split(':')[0] #Recuperation du GID du genome auquel appartient la proteine ruban a mesurer
51
52          frm = coded_by[len(coded_by)-1].split(':')[1].split('..')[0]
53
54          to = coded_by[len(coded_by)-1].split(':')[1].split('..')[1].split(')')[0] #Recuperation des positions de depart et de fin du gene codant pour la proteine
55
56          if (frm.isdigit()) and (to.isdigit()):
57
58            nuc_size_query = (int(to)-int(frm))+1 #Taille de la proteine en nucleotides
59
60            print '\nTaille de la sequence en aa : ',len(f.extract(rec.seq)),', en nuc : ',nuc_size_query #Affichage des tailles
61
62            print '>> Ecriture du genome : ',gid_coded_by,'...' #Affiche le GID du genome
63
64            query_genome = Entrez.efetch(db='nuccore', id=gid_coded_by, rettype='gbwithparts', retmode='text') #Requete au NCBI pour recuperer le genome complet
65
66            record=SeqIO.read(query_genome, "genbank")
67
68            SeqIO.write(record, 'query_file_genome', 'genbank') #Ecriture du genome au format Genbank
69
```

```
70            print '>> Fichier GB ecrit !','\n'
71
72            list_len=[]
73
74            for rec in SeqIO.parse('query_file_genome','genbank'): #↙
    Biopython decode le fichier Genbank du genome complet
75
76              for f in rec.features:
77
78                if f.type == 'CDS':
79
80                  list_len.append(len(f.extract(rec.seq))) #Stocke les ↙
    tailles des CDS dans une liste
81
82              print 'Taille du genome :', len(rec.seq),' nuc.' #Affiche la ↙
    taille du genome
83
84              print 'La liste est longue de :',len(list_len),'CDS' #Affiche↙
     le nombre de CDS dans le genome
85
86              if (len(list_len)>10): #Nous voulons plus de 10 CDS dans le ↙
    genome, sinon c est un genome incomplet
87
88                print 'Les 2 plus gros CDS sont : ',sorted(list_len,reverse↙
    =True)[0],', et : ',sorted(list_len,reverse=True)[1] #Affiche les 2 plus ↙
    gros CDS
89
90                if (nuc_size_query >= sorted(list_len,reverse=True)[1]): #↙
    Si le ruban a mesurer est plus gros ou egal au 2eme plus gros CDS
91
92                  print '*** OK TAILLE :',nuc_size_query,'nuc est plus ↙
    grand ou egal a ',sorted(list_len,reverse=True)[1],'nuc.'
93                  print '\n== Annotation validee ==\n'
94
95                  virus_valides.write('> Virus valide '+ gid) #Ecriture d'↙
    un fichier avec les GID des virus valides
96
97                  seq_tmp = open('query_file.gb','r')
98                  seq_tmp_aa = SeqIO.read(seq_tmp, 'genbank')
99                  seq_tmp_fasta = SeqIO.write(seq_tmp_aa, fasta_tmp, 'fasta↙
    ') #Ecriture de la sequence au format FASTA dans le fichier des sequences↙
     validees
100
101               else:
102
103                 print '*** TROP COURT :',nuc_size_query,'nuc est plus ↙
    petit que :',sorted(list_len,reverse=True)[1],'nuc.'
104                 print '\n== Annotation rejetee ==\n'
105         else: #Si les positions de depart et de fin ne sont pas des int, ↙
    erreur CDS
106
107           print 'Erreur CDS, Sequence a ecarter'
108
109     elif (rec.annotations['taxonomy'][0] == 'Bacteria') or (rec.annotations↙
```

```
['taxonomy'][0] == 'Archaea'): #Si l'organisme est BACTERIE
110
111         for f in rec.features:
112
113             if f.type == 'CDS':
114
115                 coded_by = f.qualifiers['coded_by'][0].split('complement(') #↙
    Recuperation du GID du genome auquel appartient la proteine ruban a ↙
    mesurer
116
117                 gid_coded_by = coded_by[len(coded_by)-1].split(':')[0]
118
119                 frm = coded_by[len(coded_by)-1].split(':')[1].split('..')[0]
120
121                 to = coded_by[len(coded_by)-1].split(':')[1].split('..')[1].split↙
    (')')[0] #Recuperation des positions de depart et de fin du gene codant ↙
    pour la proteine
122
123                 if (frm.isdigit()) and (to.isdigit()):
124
125                     nuc_size_query = (int(to)-int(frm))+1 #Taille de la proteine en↙
     nucleotides
126
127                     print '\nTaille de la sequence en aa : ',len(f.extract(rec.seq)↙
    );', en nuc : ',nuc_size_query #Affichage des tailles
128
129                     print '>> Ecriture du genome : ',gid_coded_by,'...' #Affiche le↙
     GID du genome
130
131                     query_genome = Entrez.efetch(db='nuccore', id=gid_coded_by, ↙
    rettype='gbwithparts', retmode='text') #Requete au NCBI pour recuperer le↙
     genome complet
132
133                     record=SeqIO.read(query_genome, "genbank")
134
135                     SeqIO.write(record, 'query_file_genome', 'genbank') #Ecriture ↙
    du genome au format Genbank
136
137                     print '>> Fichier GB ecrit !\n'
138
139                     list_len=[]
140
141                     frmRange = int(to)-50000
142                     toRange = int(frm)+50000 #Definit les + ou - 50000 nucleotides ↙
    autour d'un ruban a mesurer
143
144                     intFrm = int(frm)
145                     intTo = int(to)
146
147                     integrase=False #booleen utilise pour verifier la presence d'↙
    une integrase
148
149                     for rec in SeqIO.parse('query_file_genome','genbank'): #↙
    Biopython decode le fichier Genbank du genome complet
```

```
150
151             for f in rec.features:
152
153               frmI = int(f.location.start)
154               toI = int(f.location.end) #positions de chaque CDS lu
155
156               if f.type == 'CDS':
157
158                 list_len.append(len(f.extract(rec.seq))) #Stocke les tailles des CDS dans une liste
159
160                 if ('product' in f.qualifiers) and (len(f.qualifiers['product'])>0): #Lecture des annotations des CDS
161
162                   product=str(f.qualifiers['product'])
163
164                   if product.find('integrase') != -1 : #Si une annotation est INTEGRASE
165
166                     if (frmRange>0) and (((frmI > frmRange) and (toI < intFrm)) or ((frmI > intTo) and (toI < toRange))): #Si cette annotation est comprise dans les positions acceptables
167
168                       print '\n-- CDS \n Integrase detectee :',f.qualifiers['product']
169                       print 'Position :          ',frmI,'..',toI
170                       print 'Range    :',frmRange,'.....................',toRange
171                       print 'TMP      :          ',intFrm,'..',intTo
172                       integrase=True #Valider la contrainte INTEGRASE
173                       print '\n integrase =',integrase
174                       print '-------> Integrase detectee a + ou - 50 000 de TMP \n'
175
176               elif f.type == 'misc_feature': #Utilise pour rechercher dans les misc_feature
177
178                 if ('note' in f.qualifiers) and (len(f.qualifiers['note'])>0):
179
180                   note=str(f.qualifiers['note'])
181
182                   if note.find('integrase') != -1 :
183
184                     if (frmRange>0) and (((frmI > frmRange) and (toI < intFrm)) or ((frmI > intTo) and (toI < toRange))): #Si une annotation integrase est detectee dans des positions acceptables
185                       print '\n-- MISC \n Integrase detectee :',f.qualifiers['note']
186                       print 'Position :          ',frmI,'..',toI
187                       print 'Range    :',frmRange,'.....................',toRange
188                       print 'TMP      :          ',intFrm,'..',intTo
189                       integrase=True #Valider la contrainte INTEGRASE
```

```
190                     print '\n integrase =',integrase
191                     print '-------> Integrase detectee a + ou - 50 000 de TMP \n'
192
193             seuil=len(list_len)-((len(list_len)*98/100)) #Etablit le seuil pour la contrainte de TAILLE de la proteine
194
195             print 'Taille du genome :', len(rec.seq),' nuc.' #Affiche la taille du genome
196             print 'La liste est longue de :',len(list_len),'CDS' #Affiche le nombre de CDS dans le genome
197
198             if (len(list_len)>10): #Si le genome contient plus de 10 CDS
199
200               print '98% correspond aux ',seuil,' plus gros CDS. Dont le plus petit est :',sorted(list_len,reverse=True)[seuil-1],'et le plus gros ',sorted(list_len,reverse=True)[0],'\n'
201
202               if (nuc_size_query >= sorted(list_len,reverse=True)[seuil-1]) and (integrase == True): #Si les contraintes TAILLE et INTEGRASE sont validees
203
204                 print '*** OK Presence integrase car : ( integrase =',integrase,')'
205                 print '*** OK TAILLE :',nuc_size_query,'nuc est plus grand ou egal a ',sorted(list_len,reverse=True)[seuil-1],'nuc.\n'
206                 print '\n== Annotation validee ==\n'
207
208                 bacterie_valides.write('> Bacterie valide '+ gid) #Ecriture d'un fichier avec les GID des bacteries valides
209                 seq_tmp = open('query_file.gb','r')
210                 seq_tmp_aa = SeqIO.read(seq_tmp, 'genbank')
211                 seq_tmp_fasta = SeqIO.write(seq_tmp_aa, fasta_tmp, 'fasta') #Ecriture de la sequence au format FASTA dans le fichier des sequences validees
212
213               elif (nuc_size_query >= sorted(list_len,reverse=True)[seuil-1]) and (integrase == False): #Si SEULE la contrainte TAILLE est validee
214
215                 print '*** PAS de presence integrase car : ( integrase =',integrase,')'
216                 print '*** OK TAILLE :',nuc_size_query,'nuc est plus grand ou egal a ',sorted(list_len,reverse=True)[seuil-1],'nuc.\n'
217                 print '\n== Annotation douteuse ==\n'
218
219                 seq_tmp = open('query_file.gb','r')
220                 seq_tmp_aa = SeqIO.read(seq_tmp, 'genbank')
221                 SeqIO.write(seq_tmp_aa, annot_doute, 'fasta') #Ecriture de la sequence au format FASTA dans le fichier des sequences douteuses
222
223               elif (nuc_size_query < sorted(list_len,reverse=True)[seuil-1]) and (integrase == True): #Si SEULE la contrainte INTEGRASE est validee
224
```

```
225                     print '*** OK Presence integrase car : ( integrase =',integrase,')'
226                     print '*** TROP COURT :',nuc_size_query,'nuc est plus petit que :',sorted(list_len,reverse=True)[seuil-1],'nuc.\n'
227                     print '\n== Annotation douteuse ==\n'
228
229                     seq_tmp = open('query_file.gb','r')
230                     seq_tmp_aa = SeqIO.read(seq_tmp, 'genbank')
231                     SeqIO.write(seq_tmp_aa, annot_doute, 'fasta') #Ecriture de la sequence au format FASTA dans le fichier des sequences douteuses
232
233                 else: #Sinon, si AUCUNE des contraintes sont validees
234
235                     print '*** PAS de presence integrase car : ( integrase =',integrase,')'
236                     print '*** TROP COURT :',nuc_size_query,'nuc est plus petit que :',sorted(list_len,reverse=True)[seuil-1],'nuc.\n'
237                     print '\n== Annotation rejetee ==\n'
238
239                     seq_tmp = open('query_file.gb','r')
240                     seq_tmp_aa = SeqIO.read(seq_tmp, 'genbank')
241                     SeqIO.write(seq_tmp_aa, annot_rejete, 'fasta') #Ecriture de la sequence au format FASTA dans le fichier des sequences rejetees
242
243             else: #Si les positions de depart et de fin ne sont pas des int, erreur CDS
244
245                 print 'Erreur CDS, Sequence a ecarter'
246
247         else: #Si l'organisme n'est ni VIRUS, ni BACTERIE
248
249             print '>>>>> Genome autre que virus ou bacterie <<<<<<<'
250
251 virus_valides.close()
252 bacterie_valides.close()
253 fasta_tmp.close()
254 annot_doute.close()
255 annot_rejete.close() #Fermeture des fichiers
```

scripts/script_integrase.py

APPENDICE B

COMPARAISON DE SÉQUENCES

B.1 Définitions

En plus de la *distance*, vue en page 24, d'autres termes propres à la comparaison de séquences sont à définir.

— La *similarité* est la mesure de la ressemblance entre deux séquences.

— Deux séquences sont *homologues* si elles dérivent d'un ancêtre commun.

Notons que la similarité n'implique pas obligatoirement l'homologie. Par exemple, deux séquences grandement similaires ne découlent pas obligatoirement d'un ancêtre commun (c'est souvent le cas des très courtes séquences). La similarité peut alors simplement être due au hasard.

Inversement, l'homologie n'impliquera pas toujours une similarité élevée. Il peut y avoir des séquences homologues considérablement distantes. Ça sera le cas dans notre étude des protéines *ruban à mesurer*, où des séquences homologues présentent peu de similarité, mais seuls des endroits clés ont été conservés durant l'évolution.

B.2 Mutations

Nous avons vu précédemment que des modifications peuvent survenir sur une séquence d'ADN. Parmi elles, nous avons déjà cité la substitution qui est la modification d'une base en une autre.

Rappel sur la substitution : ATGATG $\xrightarrow{mute\ en}$ ATAATG

Il existe également deux autres types de mutations :

1. L'**insertion** est l'ajout d'une base

$$\text{ATGATG} \xrightarrow{mute\ en} \text{ATGAATG}$$

2. La **délétion** ou, **suppression** est la perte d'une base.

$$\text{ATGATG} \xrightarrow{mute\ en} \text{ATATG}$$

Ces mutations sont des renseignements cruciaux pour reconstruire l'histoire des duplications. Nous allons donc voir dans la prochaine section comment comparer ces duplications pour trouver les plus récentes en tenant compte de ces mutations.

B.3 Alignement

Pour calculer la distance entre les duplications, nous avons vu qu'il faut comparer les séquences dupliquées. Lorsque ces séquences deviennent complexes, la comparaison devient plus difficile et on aura alors besoin d'*aligner* ces séquences.

Un *alignement* est la superposition des lettres des deux séquences. Ces lettres peuvent être des nucléotides ou bien des acides aminés.

On peut volontairement créer des brèches dans un alignement. Ces brèches sont appelées des *gaps*, symbolisées par le caractère : –. Deux brèches ne sont généralement pas alignées ensemble.

Deux exemples d'alignements présentant des brèches :

```
GTAGCTA          CGATGACT
GTA-CTA          C--T'GA-T
```

Ces exemples présentent des alignements parfaits entre nucléotides (`A-A`, `T-T`, `G-G`, `C-C`). Cependant, il est possible d'aligner n'importe quoi avec n'importe quoi.

Par exemple,
```
ACGTCT
GCATTG
```
est aussi un alignement.

C'est certes un alignement, mais de moins bonne qualité. Certains alignements sont meilleurs que d'autres et, pour avoir une idée claire de cela, il faut leur attribuer des scores.

B.4 Score des alignements

B.4.1 Matrice de score

Considérons notre alphabet de nucléotides $\mathcal{A}$ auquel nous ajoutons le symbole de gap « - »

$$\mathcal{A} = \{\mathtt{A},\ \mathtt{C},\ \mathtt{G},\ \mathtt{T}\}$$
$$\mathcal{A}' = \mathcal{A} \cup \{\mathtt{-}\}$$

Considérons $x, y \in \mathcal{A}'$ avec $s(x, y)$ représentant le score de l'alignement de x avec y. Ces scores sont donnés par exemple dans la matrice suivante.

$$s(\mathtt{A},\mathtt{A}) = +1$$
$$s(\mathtt{A},\mathtt{G}) = -1$$
$$s(\mathtt{A},\mathtt{G}) = s(\mathtt{G},\mathtt{A})$$
$$s(\mathtt{A},\mathtt{-}) = -2$$

$$\begin{bmatrix} & \mathtt{A} & \mathtt{C} & \mathtt{G} & \mathtt{T} & \mathtt{-} \\ \mathtt{A} & +1 & -1 & -1 & -1 & -2 \\ \mathtt{C} & -1 & +1 & -1 & -1 & -2 \\ \mathtt{G} & -1 & -1 & +1 & -1 & -2 \\ \mathtt{T} & -1 & -1 & -1 & +1 & -2 \\ \mathtt{-} & -2 & -2 & -2 & -2 & \times \end{bmatrix}$$

Le score d'un alignement est la somme des scores de ses colonnes. Le meilleur alignement est celui qui possède le score le plus élevé.

Un exemple est présenté ci-dessous.

AC-CGT	CAGGTC	-ACCGT
ACACGT	G-C-C-	ACACGT
$s = 3$	$s = -9$	$s = -1$

B.4.2 Distance d'édition

Les exemples précédents pénalisent les substitutions et récompensent les égalités. Cependant, le calcul de la *distance d'édition* fait l'inverse. Elle repose sur une matrice de coûts qui attribue un point à chaque différence dans l'alignement.

Ainsi, comme pour la distance de Hamming, plus le coût est élevé, moins l'alignement sera de bonne qualité. La distance de Hamming repose sur la matrice suivante :

$$\begin{bmatrix} & \mathtt{A} & \mathtt{C} & \mathtt{G} & \mathtt{T} & \mathtt{-} \\ \mathtt{A} & 0 & +1 & +1 & +1 & +1 \\ \mathtt{C} & +1 & 0 & +1 & +1 & +1 \\ \mathtt{G} & +1 & +1 & 0 & +1 & +1 \\ \mathtt{T} & +1 & +1 & +1 & 0 & +1 \\ \mathtt{-} & +1 & +1 & +1 & +1 & \times \end{bmatrix}$$

A l'aide de cette matrice, le coût d'un alignement sera nul uniquement si les deux séquences sont strictement identiques.

Ci-dessous, un exemple du calcul de la distance d'édition sur des alignements de `ATGACTG` avec `CTAGTG` :

`ATGACTG`	`ATGACTG`	`ATGACTG`
`CTAGTG-`	`CTAG-TG`	`CT-AGTG`
$c = 6$	$c = 4$	$c = 3$

APPENDICE C

FORMATS BIOINFORMATIQUES

C.1 Format FASTA

Le format FASTA est utilisé afin de présenter des séquences de nucléotides ou d'acides aminés. Il est devenu le standard que la majorité des utilitaires bioinformatiques utilisent. Il se caractérise par une première ligne commençant par le signe supérieur ">" directement suivi de la description de la séquence. On retrouve généralement l'identificateur et la description de la séquence ainsi que l'espèce à laquelle elle est rattachée. Les lignes suivantes contiennent la séquence et elles ont généralement une longueur de 70 caractères. Ce standard permet de facilement manipuler les séquences par exemple avec l'utilitaire **grep**. Un exemple de séquence au format FASTA est présenté ci-dessous, il illustre une séquence d'acides aminés annotée *tape measure*, dans la bactérie *Escherichia Coli* :

C.2 Format GenBank

En plus de donner la séquence biologique comme le format FASTA, une fiche GenBank dispose d'informations supplémentaires. Un exemple de fichier GenBank est proposé

```
>gi|320652172|gb|EFX20483.1| tape measure protein [Escherichia coli]
MSQPVGDLVIDLSLDAVRFDEQMSRVRRHFSGLDTDVRKTASAVEQGLSRQALAAQKAGISVGQYKAAMR
TLPAQFTDIATQLAGGQNPWLILLQQGGQVKDSFGGMIPMFRGLAGAITLPMVGVTSLAVATGALVYAWY
QGDSTLSAFNKTLVLSGNQSGLTADRMLTLSRAGQAAGLTFNQARESLAALVNAGVRGGEQFDAINQSVA
SFCFCIRCG
```

ci-dessous.

On retrouve, par exemple, la taxonomie (`ORGANISM Escherichia coli O157:H- str. H 2687 Bacteria ...`) mais également les caractéristiques (`FEATURES`) de la séquence. Parmi ces caractéristiques, nous avons beaucoup utilisé la partie codante (`CDS`). Celle-ci renseigne sur le génome (et la portion de génome) qui code cette protéine. Ici, la protéine *tail length tape measure protein, partial [Escherichia coli O157 :H-str. H 2687]* est codée par le génome dont le numéro d'accession est AETZ01000030.1.

```
LOCUS       EFX20483                219 aa            linear   BCT 31-JAN-2011
DEFINITION  tail length tape measure protein, partial [Escherichia coli O157:H-
            str. H 2687].
ACCESSION   EFX20483
VERSION     EFX20483.1  GI:320652172
DBSOURCE    accession AETZ01000030.1
KEYWORDS    .
SOURCE      Escherichia coli O157:H- str. H 2687
ORGANISM    Escherichia coli O157:H- str. H 2687
            Bacteria; Proteobacteria; Gammaproteobacteria; Enterobacteriales;
            Enterobacteriaceae; Escherichia.
COMMENT     Method: conceptual translation.
FEATURES             Location/Qualifiers
     source          1..219
                     /organism="Escherichia coli O157:H- str. H 2687"
                     /strain="H 2687"
                     /serovar="O157:H-"
                     /db_xref="taxon:926028"
     Protein         1..>219
                     /product="tail length tape measure protein"
     Region          1..>212
                     /region_name="COG5281"
                     /note="Phage-related minor tail protein [Function
                     unknown]"
                     /db_xref="CDD:227606"
     Region          49..>212
                     /region_name="TMP_2"
                     /note="Prophage tail length tape measure protein;
                     pfam06791"
                     /db_xref="CDD:148412"
     CDS             1..219
                     /locus_tag="ECO2687_16291"
                     /coded_by="complement(AETZ01000030.1:<1..657)"
                     /note="COG5281 Phage-related minor tail protein"
                     /transl_table=11
ORIGIN
        1 msqpvgdlvi dlsldavrfd eqmsrvrrhf sgldtdvrkt asaveqglsr qalaaqkagi
       61 svgqykaamr tlpaqftdia tqlaggqnpw lillqqggqv kdsfggmipm frglagaitl
      121 pmvgvtslav atgalvyawy qgdstlsafn ktlvlsgnqs gltadrmltl sragqaaglt
      181 fnqareslaa lvnagvrgge qfdainqsva sfcfcircg
//
```

C.3 ClustalW2

En bioinformatique, les outils d'alignement de séquences sont grandement utilisés. À la différence de BLAST, ClustalW est un programme d'alignement global.

On utilise l'alignement global dans le cas où les séquences sont de taille assez proche. Le résultat final est un alignement sur l'ensemble des séquences et comportera sûrement des brèches. ClustalW est un programme d'alignement de séquences multiples permettant ainsi d'aligner globalement plusieurs séquences ensemble. Ci-dessous un exemple d'alignement retourné par ClustalW. Les séquences SEQ1 et SEQ2 ont été soumises en entrée et, en sortie, un alignement global est présenté à la suite.

En entrée :

```
>SEQ1
WGKLNFEEKKAVLYSNTPEVMAETMLNLGL
>SEQ2
WKDYQPQVKELKAKNQSFLDVLSQSQDKIVH
```

En sortie :

```
CLUSTAL 2.1 multiple sequence alignment

SEQ1            WGKLNFEEKKAVLYSNTP---EVMAETMLNLGL 30
SEQ2            WK--DYQPQVKELKAKNQSFLDVLSQSQDKIVH 31
                *   ::: :   * ::.    :*:::: ::
```

BIBLIOGRAPHIE

Ackermann, H.-W. 2011. « Bacteriophage taxonomy ». *Microbiology Australia*, vol. 32, no. 2, p. 90–94.

Alberts, B. 1995. *Biologie moléculaire de la cellule.* Paris : Flammarion Médecine-Sciences, 3ième édition.

Altschul, S. F., W. Gish, W. Miller, E. W. Myers et D. J. Lipman. 1990. « Basic local alignment search tool ». *Journal of molecular biology*, vol. 215, no. 3, p. 403–410.

Bailey, T., et C. Elkan. 1994. « Fitting a mixture model by expectation maximization to discover motifs in biopolymers. ». *Proceedings of International Conference on Intelligent Systems for Molecular Biology*, vol. 2, p. 28–36.

Belcaid, M., A. Bergeron et G. Poisson. 2011. « The evolution of the tape measure protein : units, duplications and losses ». *BMC bioinformatics*, vol. 12 (Suppl 9), no. S10, p. 1–12.

Benson, G. 1999. « Tandem repeats finder : a program to analyze dna sequences ». *Nucleic acids research*, vol. 27, no. 2, p. 573–580.

Beveridge, T. J., et L. L. Graham. 1991. « Surface layers of bacteria ». *Microbiological reviews*, vol. 55, no. 4, p. 684–705.

Blattner, F. R., G. Plunkett, C. A. Bloch, N. T. Perna, V. Burland, M. Riley, J. Collado-Vides, J. D. Glasner, C. K. Rode, G. F. Mayhew *et al.* 1997. « The complete genome sequence of escherichia coli k-12 ». *Science*, vol. 277, no. 5331, p. 1453–1462.

Breitbart, M., P. Salamon, B. Andresen, J. M. Mahaffy, A. M. Segall, D. Mead, F. Azam et F. Rohwer. 2002. « Genomic analysis of uncultured marine viral communities ». *Proceedings of the National Academy of Sciences*, vol. 99, no. 22, p. 14250–14255.

Buchen-Osmond, C. 2003. « The universal virus database ictvdb ». *Computing in Science Engineering*, vol. 5, no. 3, p. 16–25.

Cantalupo, P. G., B. Calgua, G. Zhao, A. Hundesa, A. D. Wier, J. P. Katz, M. Grabe, R. W. Hendrix, R. Girones, D. Wang *et al.* 2011. « Raw sewage harbors diverse viral populations ». *mBio*, vol. 2, no. 5, p. 00180–11.

Chapman, B., et J. Chang. 2000. « Biopython : Python tools for computational biology ». *ACM SIGBIO Newsletter*, vol. 20, no. 2, p. 15–19.

Cock, P. J., T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski *et al.* 2009. « Biopython : freely available python tools for computational molecular biology and bioinformatics ». *Bioinformatics*, vol. 25, no. 11, p. 1422–1423.

Curtis, T. P., W. T. Sloan et J. W. Scannell. 2002. « Estimating prokaryotic diversity and its limits ». *Proceedings of the National Academy of Sciences*, vol. 99, no. 16, p. 10494–10499.

d'Hérelle, F. 1917. « Sur un microbe invisible antagoniste des bacilles dysentériques ». *Comptes rendus de l'académie des Sciences de Paris*, vol. 165, p. 373–375.

Elemento, O., O. Gascuel et M.-P. Lefranc. 2002. « Reconstructing the duplication history of tandemly repeated genes ». *Molecular Biology and Evolution*, vol. 19, no. 3, p. 278–288.

Elzanowski, A., et J. Ostell. 2012. « The genetic codes ». En ligne. <`http://www.ncbi.nlm.nih.gov/Taxonomy/Utils/wprintgc.cgi`>. Consulté le 24 Septembre 2012.

Fitch, W. M. 1971. « Toward defining the course of evolution : minimum change for a specific tree topology ». *Systematic Zoology*, vol. 20, no. 4, p. 406–416.

Hamming, R. W. 1950. « Error detecting and error correcting codes ». *Bell System technical journal*, vol. 29, no. 2, p. 147–160.

Hemphill, H., et H. Whiteley. 1975. « Bacteriophages of bacillus subtilis ». *Bacteriological reviews*, vol. 39, no. 3, p. 257–315.

Jaynes, E. T. 1957. « Information theory and statistical mechanics ». *Physical review*, vol. 106, no. 4, p. 620.

Katsura, I., et R. W. Hendrix. 1984. « Length determination in bacteriophage lambda tails ». *Cell*, vol. 39, no. 3, p. 691–698.

Lajoie, M., D. Bertrand, N. El-Mabrouk et O. Gascuel. 2007. « Duplication and inversion history of a tandemly repeated genes family ». *Journal of Computational Biology*, vol. 14, no. 4, p. 462–478.

Larkin, M., G. Blackshields, N. Brown, R. Chenna, P. McGettigan, H. McWilliam, F. Valentin, I. Wallace, A. Wilm, R. Lopez, J. Thompson, T. Gibson et D. Higgins. 2007. « Clustal w and clustal x version 2.0 ». *Bioinformatics*, vol. 23, no. 21, p. 2947–2948.

Li, W., et A. Godzik. 2006. « Cd-hit : a fast program for clustering and comparing large sets of protein or nucleotide sequences ». *Bioinformatics*, vol. 22, no. 13,

p. 1658–1659.

Novotny, V., Y. Basset, S. E. Miller, G. D. Weiblen, B. Bremer, L. Cizek et P. Drozd. 2002. « Low host specificity of herbivorous insects in a tropical forest ». *Nature*, vol. 416, no. 6883, p. 841–844.

Pell, L. G., V. Kanelis, L. W. Donaldson, P. Lynne Howell et A. R. Davidson. 2009. « The phage lambda major tail protein structure reveals a common evolution for long-tailed phages and the type vi bacterial secretion system ». *Proceedings of the National Academy of Sciences*, vol. 106, no. 11, p. 4160–4165.

Rivals, E. 2004. « A survey on algorithmic aspects of tandem repeats evolution ». *International Journal of Foundations of Computer Science*, vol. 15, no. 02, p. 225–257.

Rohwer, F. 2003. « Global phage diversity ». *Cell*, vol. 113, no. 2, p. 141.

Sanger, F., G. M. Air, B. G. Barrell, N. L. Brown, A. R. Coulson, J. C. Fiddes, C. A. Hutchison, P. M. Slocombe et M. Smith. 1977. « Nucleotide sequence of bacteriophage phi x174 dna ». *Nature*, vol. 265, no. 5596, p. 687–695.

Santos, M. A. S., T. Ueda, K. Watanabe et M. F. Tuite. 1997. « The non-standard genetic code of candida spp. : an evolving genetic code or a novel mechanism for adaptation ? ». *Molecular Microbiology*, vol. 26, no. 3, p. 423–431.

Schneider, T. D., et R. M. Stephens. 1990. « Sequence logos : a new way to display consensus sequences ». *Nucleic acids research*, vol. 18, no. 20, p. 6097–6100.

Siponen, M., G. Sciara, M. Villion, S. Spinelli, J. Lichière, C. Cambillau, S. Moineau et V. Campanacci. 2009. « Crystal structure of orf12 from lactococcus lactis phage p2 identifies a tape measure protein chaperone ». *Journal of bacteriology*, vol. 191, no. 3, p. 728–734.

Srividhya, K., G. Rao, L. Raghavenderan, P. Mehta, J. Prilusky, S. Manicka, J. Sussman et S. Krishnaswamy. 2006. *Database and Comparative Identification of Prophages*. Coll. Huang, D.-S., K. Li et G. Irwin, éditeurs, Coll. « *Intelligent Control and Automation* ». T. 344, série *Lecture Notes in Control and Information Sciences*, p. 863–868. Springer Berlin Heidelberg.

Twort, F. 1915. « An investigation on the nature of ultra-microscopic viruses ». *The Lancet*, vol. 186, no. 4814, p. 1241–1243.

U.S. National Library of Medicine. 2012. « National center for biotechnology information ». En ligne. <`http://www.ncbi.nlm.nih.gov/`>. Consulté le 24 Septembre 2012.

Villarreal, L. P. 2004. « Are viruses alive ? ». *Scientific American*, vol. 291, no. 6, p. 100–105.

Wagner, W. H. 1961. « Problems in the classification of ferns ». *Recent advances in botany*, vol. 1, p. 841–844.

Wang, I.-N., D. L. Smith et R. Young. 2000. « Holins : the protein clocks of bacteriophage infections ». *Annual Reviews in Microbiology*, vol. 54, no. 1, p. 799–825.

Whitman, W. B., D. C. Coleman et W. J. Wiebe. 1998. « Prokaryotes : The unseen majority ». *Proceedings of the National Academy of Sciences*, vol. 95, no. 12, p. 6578–6583.

Wyatt, P. J. 1970. « Cell wall thickness, size distribution, refractive index ratio and dry weight content of living bacteria (stapylococcus aureus) ». *Nature*. no. 226, p. 277–279.

Zhang, Z., S. Schwartz, L. Wagner et W. Miller. 2000. « A greedy algorithm for aligning dna sequences ». *Journal of Computational biology*, vol. 7, no. 1-2, p. 203–214.

Zhou, Y., Y. Liang, K. H. Lynch, J. J. Dennis et D. S. Wishart. 2011. « Phast : A fast phage search tool ». *Nucleic Acids Research*, vol. 39, no. S2, p. W347–W352.

Zuber, B., M. Haenni, T. Ribeiro, K. Minnig, F. Lopes, P. Moreillon et J. Dubochet. 2006. « Granular layer in the periplasmic space of gram-positive bacteria and fine structures of enterococcus gallinarum and streptococcus gordonii septa revealed by cryo-electron microscopy of vitreous sections ». *Journal of bacteriology*, vol. 188, no. 18, p. 6652–6660.